2011-09

A registrar administration system requirements analysis and product recommendation for Marine Corps University, Quantico, VA

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

A REGISTRAR ADMINISTRATION SYSTEM REQUIREMENTS ANALYSIS AND PRODUCT RECOMMENDATION FOR MARINE CORPS UNIVERSITY, QUANTICO, VA

by

Daniel E. Good

September 2011

Thesis Co-Advisors: Luqi
Karl Pfeiffer

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2011	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE A Registrar Administration System Requirements Analysis and Product Recommendation for Marine Corps University, Quantico, VA		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Daniel E. Good			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) Marine Corps University, 2076 South Street, Quantico, VA 22134		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ___N/A_____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE** A	

13. ABSTRACT (maximum 200 words)

Marine Corps University (MCU) is a relatively young organization and continues to mature as it brings more academic functionality and oversight under centralized control, especially in the area of Information Technology (IT). Much of MCU's IT control and responsibility still remains decentralized down to the school and college level. This research focuses on a specific IT capability, a Registrar Administration System (RAS). An RAS may also be termed a Student Information System (SIS). This type of system performs many functions. Some of them typically include the ability to hold or access personal student and faculty information; correlate students to courses completed, grades received and when; provide faculty a portal to upload course grades; and provide the Registrar's office access to generate transcripts. It may also include functionality for Registrar, course scheduling, or alumni needs. In this research, we conduct a requirements analysis (RA) to determine MCU's needs for this type of system.

After understanding MCU's requirements, we conduct a market analysis to learn about systems that are being employed at institutions similar to MCU. Next, product characteristics, or factors, to be considered and Likert rating scales are defined in preparation for an evaluation of each system. We conduct a product comparison based on our system evaluations and conclude by recommending the best system for MCU.

14. SUBJECT TERMS USMC, MCU, Student Information System, IET Master Plan, requirements analysis, software engineering, market analysis, product comparison, product recommendation, business study			**15. NUMBER OF PAGES** 101
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

A REGISTRAR ADMINISTRATION SYSTEM REQUIREMENTS ANALYSIS AND PRODUCT RECOMMENDATION FOR MARINE CORPS UNIVERSITY, QUANTICO, VA

Daniel E. Good
Captain, United States Marine Corps
B.S., United States Naval Academy, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE
and
MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT

from the

NAVAL POSTGRADUATE SCHOOL
September 2011

Author: Daniel E. Good

Approved by: Luqi
Thesis Co-Advisor

Karl Pfeiffer
Thesis Co-Advisor

Peter Denning
Chair, Department of Computer Science

Dan Boger
Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Marine Corps University (MCU) is a relatively young organization and continues to mature as it brings more academic functionality and oversight under centralized control, especially in the area of Information Technology (IT). Much of MCU's IT control and responsibility still remains decentralized down to the school and college level. This research focuses on a specific IT capability, a Registrar Administration System (RAS). An RAS may also be termed a Student Information System (SIS). This type of system performs many functions. Some of them typically include the ability to hold or access personal student and faculty information; correlate students to courses completed, grades received and when; provide faculty a portal to upload course grades; and provide the Registrar's office access to generate transcripts. It may also include functionality for Registrar, course scheduling, or alumni needs. In this research, we conduct a requirements analysis (RA) to determine MCU's needs for this type of system.

After understanding MCU's requirements, we conduct a market analysis to learn about systems that are being employed at institutions similar to MCU. Next, product characteristics, or factors, to be considered and Likert rating scales are defined in preparation for an evaluation of each system. We conduct a product comparison based on our system evaluations and conclude by recommending the best system for MCU.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ASP	Active Server Page
AU	Air University
AWC	Army War College
CJCS	Chairman of the Joint Chiefs of Staff
COA	Course of Action
CSC	Command and Staff College
DB	Database
DES	Data Enterprise System
DoD	Department of Defense
EPME	Enlisted Professional Military Education
EWS	Expeditionary Warfare School
FPME	Foreign Professional Military Education
GRC	General Alfred M. Gray Marine Corps Research Center
I/O	Input/Output
IET	Information and Education Technology
IT	Information Technology
JPME	Joint Professional Military Education
LLI	Lejeune Leadership Institute
MAGTF	Marine Air Ground Task Force
MCU	Marine Corps University
MCWAR	Marine Corps War College
MOU	Memorandum of Understanding
NDU	National Defense University

NMMC	National Museum of the Marine Corps
NPS	Naval Postgraduate School
NWC	Naval War College
PAJE	Process for the Accreditation of Joint Education
PII	Personally Identifiable Information
PME	Professional Military Education
RA	Requirements Analysis
RAS	Registrar Administration System
RFP	Request for Proposal
SAW	School of Advanced Warfighting
SIS	Student Information System
SOML	School of MAGTF Logistics
TECOM	Training and Education Command
USMC	United States Marine Corps

ACKNOWLEDGMENTS

First, to my wife, Maryn, you have selflessly served our family while I was away studying and working. Your patience, understanding, and resilience helped me to work through the tough times without concern about the home front. Your leadership with Violet brought me joy as I arrived home each day to enthusiastic greetings, hugs, and kisses. Violet, you brought joy to my heart each day; playing with you was always a welcome break from studying. Lauren, thank you for your help editing my first draft. To family and friends that kept me in their prayers, thank you.

To Mr. William Wright for your vision for MCU and the IET Directorate, thank you for bringing me to MCU to discuss potential research topics, which planted the seed for this study. Throughout this endeavor, your input and feedback have been critical to the progress and relevance of this study. Thank you for investing your time and effort.

To Luis Velazquez who preceded me at NPS (2000), advised me for my capstone project at the Naval Academy (2002), and has remained a mentor ever since. Thank you for your advice and support over dinner during my thesis travel to MCU. Your ability to see the potential for this work and articulate it watered the seed for this study. Your example to me, both personal and professional, remains encouraging and motivating.

Next, I would like to thank Richard Jaques for your forthrightness, humility, and enthusiasm to discuss the topic. Your keen awareness of the right way to conduct business was very helpful as this study took shape.

To Mike Andersen, your interest in this topic and willingness to meet and discuss ideas greatly helped to focus the scope of this work. I would also like to thank Keith Jones for the illuminating and helpful conversations “over a beer” in the Trident Room.

Next, I would like to acknowledge the joint efforts of Dr. Anthony Ciavarelli, Dr. Gary Ohls and Dr. Casey Lucius for their advice, which helped me to craft the user survey referenced here with more clarity.

To Peter Denning and Craig Martell for your instruction during CS4900/4901 at the beginning of this venture. Your wisdom and advice to start early, keep a notebook, take notes on everything you read, consider using the technique of mind mapping to capture concepts and thoughts, make a schedule and keep it, manage your time and your advisors, and to keep your commitments have been invaluable to me. These are lifelong tools you have taught me. Thank you for your willingness to teach those that come along after you.

To Pam Silva and Janice Rivera in the Thesis Processing Office, thank you for your attention and patience with yet another NPS student passing through this institution. Your commitment to the presentation of the work flowing from NPS is to be commended.

To my co-advisors, Professor Luqi and Dr. Karl Pfeiffer, thank you for your trust and confidence throughout. You allowed me to work independently and provided rudder steer to keep me on course. Your guidance during our meetings was always helpful and given with a smile and the utmost professionalism. It has been a joy to work with you.

Finally, to my Lord and Savior Jesus Christ, upon whom I put my hope and faith, which You have given me by Your sovereign, undeserved grace and mercy, may You be glorified through my effort here. All that I have comes from You. You alone are wise and good. You created the heavens, the earth, and the sea and all that is in them. All things were created by You, through You, and for You…and in You all things hold together. To Christ alone, be all glory, honor, and praise, both now and forevermore. Amen. (James 1:17, Romans 16:27, Exodus 20:11, Nehemiah 9:6, Psalm 146:6, Genesis 1-2, Colossians 1:16, John 1:3)

Sola Scriptura: The Scripture alone is the standard.

Solo Christo: By Christ's work alone are people saved.

Sola Gratia: Salvation is by grace alone.

Sola Fide: Justification is by faith alone.

Soli Deo Gloria: For the glory of God alone.

I. INTRODUCTION

Marine Corps University (MCU) is a relatively young organization and continues to mature as it brings more academic functionality and oversight under centralized control, especially in the area of Information Technology (IT). Much of MCU's IT control and responsibility still remains decentralized down to the school and college level. This research focuses on a specific IT capability, a Registrar Administration System (RAS). A RAS may also be termed a Student Information System (SIS). This type of system performs many functions. Some of them typically include the ability to hold or access personal student and faculty information; correlating students to courses completed, grades received and when; providing faculty a portal to upload course grades; and providing the Registrar's office access to generate transcripts. It may also include functionality for Registrar, course scheduling, or alumni needs. In this research, we conduct a requirements analysis (RA) to determine MCU's needs for this type of system.

After understanding MCU's requirements, we conduct a market analysis to learn about systems that are being employed at institutions similar to MCU. Next, product characteristics, or factors, to be considered and Likert rating scales are defined in preparation for an evaluation of each system. We conduct a product comparison based on our system evaluations and conclude by recommending the best system to MCU.

A. MARINE CORPS UNIVERSITY

1. Historical Background

MCU is the United States Marine Corps' (USMC) organizational head of six Professional Military Education (PME) colleges and schools. They are Marine Corps War College (MCWAR); School of Advanced Warfighting (SAW); Command and Staff College (CSC); Expeditionary Warfare School (EWS); Enlisted Professional Military Education (EPME); and School of Marine Air Ground Task Force (MAGTF) Logistics (SOML), see Figure 1. Other programs under MCU are the Lejeune Leadership Institute (LLI), Commandant of the Marine Corps Fellows Program, Foreign Professional Military

Education (FPME), the Olmsted Scholar Program, the General Alfred M. Gray Marine Corps Research Center (GRC), the Marine Corps' History Division, and National Museum of the Marine Corps (NMMC) [1].

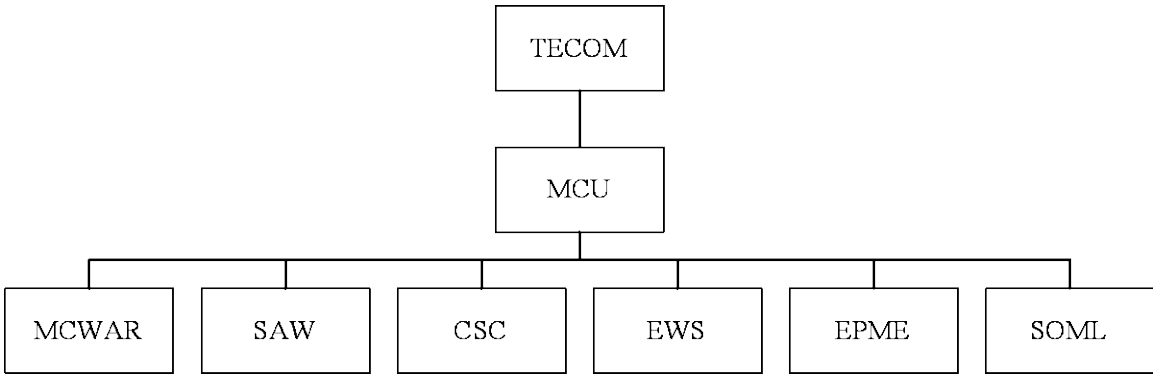

Figure 1. MCU Organizational Hierarchy

The Marine Corps' Training and Education Command (TECOM) was established in 2000 and serves as MCU's higher headquarters [2].

The schools and colleges that make up MCU have a long and independent history with origins dating back to 1891. As an organizational head, MCU was founded as recently as 1989, and has been evolving ever since. In 1999, 2001, and 2003, CSC, MCWAR, and SAW were accredited, respectively, by the Commission on Colleges of the Southern Association of Colleges and Schools to award master's degrees to students as follows [1]:

- MCWAR – Master of Strategic Studies
- SAW – Master of Operational Studies
- CSC – Master of Military Studies

MCU's SAW and CSC have also been accredited by Chairman of the Joint Chiefs of Staff (CJCS) through the Process for the Accreditation of Joint Education (PAJE) to grant Joint Professional Military Education (JPME) Phase 1 (CSC) and JMPE Phase 2 (MCWAR) credit to its graduating students [1].

MCU is a relatively young organization and continues to mature as it brings more academic functionality and oversight under centralized control, especially in the area of IT. Much of MCU's IT control and responsibility still remains decentralized down to the school and college level. In 2006, a study group was formed to determine whether the USMC's Officer PME was effectively preparing its officers to meet 21st century challenges. That study concluded that "change is the order of the day, and Professional Military Education Institutions of the Marine Corps must keep pace [3]." In 2007, after reviewing recommendations about Information and Education Technology (IET), the President of MCU commissioned a study to explore MCU's current and future IET requirements. The resulting report of this study is named Marine Corps University Information & Education Technology Master Plan and was completed in July 2008 [4]. The MCU IET Master Plan addresses several areas, many of which are outside the scope of our research; however it specifically draws attention to the need for centralized control of certain MCU functions that are redundantly or inefficiently executed. As MCU develops as an organization, it must make progress toward standardizing or centralizing redundant functions. Broad categories of such functions, as identified in the Master Plan, are administrative tracking, curriculum development, content delivery, infrastructure and network management, and IT procurement [4].

2. MCU Way Ahead

The 2008 IET Master Plan illuminates areas of concern mentioned to improve unity of command in order 1) to ensure information authority, consistency, and quality and 2) to maximize standardization, economy, and efficiency. To help achieve this end state, the IET Master Plan specifically calls for follow-on studies on unity of command issues identified therein. This research is precisely that. It is a follow-on study specifically focusing on unity of command or centralization as it relates to MCU's Registrar administration functionality [4].

3. Why This Research?

a. IET Master Plan Task 4 Recommendations

This research augments the IET Master Plan's recommendations to Task 4 that asks the study group to "identify the requirements and assess current and emerging technologies that support both administrative and academic functions within MCU to include but not limited to curriculum design, development and delivery, student registry and data base management, library and archival support, institutional research and effectiveness, publication and dissemination of scholarly research and museum asset management [4]."

b. IET Master Plan Findings

This research specifically addresses IET Master Plan findings F-31-b and F-31-c. Finding F-31-b states, "Currently, the MCU Registrar performs grade tracking through file transfer and 'cut-and-paste' and data re-entry [4]." Finding F-31-c states, "MCU needs the ability for faculty members to enter grades into a formal system themselves without the MCU Registrar being a go-between; the current system has too many people in the process, which increases the opportunity for error [4]."

c. MCU Strategic Goal 3

This research partially addresses MCU Strategic Goal 3 Objectives 3.a, 3.b, 3.c, and 3.d. Objective 3.a states, "Staff and resource the IET directorate to implement the IET Master Plan and provide comprehensive IET support to the University [4]." Objective 3.b states, "Implement systems to improve the business processes at the University and enable the free flow information [4]." Objective 3.c states, "Implement systems to facilitate curriculum development and delivery process and provide the means for increased collaboration and coordination [4]." Objective 3.d states, "Develop and implement an MCU enterprise architecture that provides for the effective operation of University systems and provides student, faculty, and staff the technology resources necessary to enhance the education processes [4]."

B. REQUIREMENTS ANALYSIS (RA)

A RAS that automates the recording of course grades and facilitates production of official academic transcripts and reports is a good step in the right direction to correct findings F-31-b and F-31-c.

Chapter II will lay a foundation for RA. Here, we give attention to methods that describe one way to conduct RA. Specifically, we discuss how to develop the initial problem statement, environment model, goals, goal hierarchy, and constraints.

In Chapter III we put these methods into practice by conducting a RA for our customer, MCU. We develop a set of requirements based on the needs of MCU for such a RAS. We compose an initial problem statement, environment model, goals, and constraints. We build high-level use cases, use design notation and context diagrams, and examine external interfaces to help us gain a better understanding of the problem domain. We conduct a user survey to gather input across the range of users. We consider this input and communicate with MCU to refine our goals, constraints, and environment model. Chapter III concludes with a completed goal hierarchy and list of constraints. After we have established MCU's requirements, we will conduct a business study to evaluate and compare products currently in use.

C. MARKET ANALYSIS AND PRODUCT COMPARISON

In Chapter IV, we conduct a market analysis and product evaluation. We start with a market analysis to learn about the systems that are being employed at institutions similar to MCU. We discuss various evaluation and comparison methods and then choose one of them to employ for our own use. Next, we define the product characteristics, or factors, we will consider and Likert rating scales we intend to use in our evaluation. We conclude Chapter IV by evaluating four candidate software products.

In Chapter V, we conduct a product comparison that is based on the evaluations we conducted in Chapter IV. We start by mapping our Likert rating scales to numerical values. Next, we define three most probable scenarios in which MCU might find itself in the near future. We establish weighting values for each factor based upon each of the

three scenarios. We then bring our product evaluations, scenarios, and weighting values together to compare the products by calculating weighted sums. For each scenario, the product with the largest weighted sum is then recommended to MCU as the best product for that given scenario.

Chapter VI concludes this study with a summary and recommendations for future work.

II. REQUIREMENTS ANALYSIS

This chapter introduces some of the relevant building blocks useful for conducting requirements analysis (RA). The need for requirements analysis is explained, as are some general concepts to illustrate structure, derivation, and procedures. Several components will be defined and illustrated with diagrams, and goal development will be described in greater detail. This chapter is not meant to be a comprehensive survey of all aspects for consideration during RA; rather, our purpose is to lay a foundation on which Chapter III rests.

A. FOUNDATIONS

The purpose of RA is to understand the needs of the customer in sufficient detail such that a software system that meets those needs may be built. A customer paying for development of a software system is usually an organization that will use or sell the system [5]. Throughout the process of RA, the primary focus is on capturing *what* the system must do, not *how* it should do it [6].

1. Structure of Requirements Analysis

RA often starts with the customer writing a short informal problem statement to describe the basic need. Software analysts abstract, derive, and piece together information from the customer to write a coherent description of the problem, called a requirements document. Generating a sufficiently complete set of requirements takes much effort to gain the necessary information from the customer and may require gathering knowledge from experts on the problem area. The analysts must be able to detect inconsistencies, missing information, and ensure the requirements meet the actual need of the customer [5]. Figure 2 shows the components of RA and how they are related.

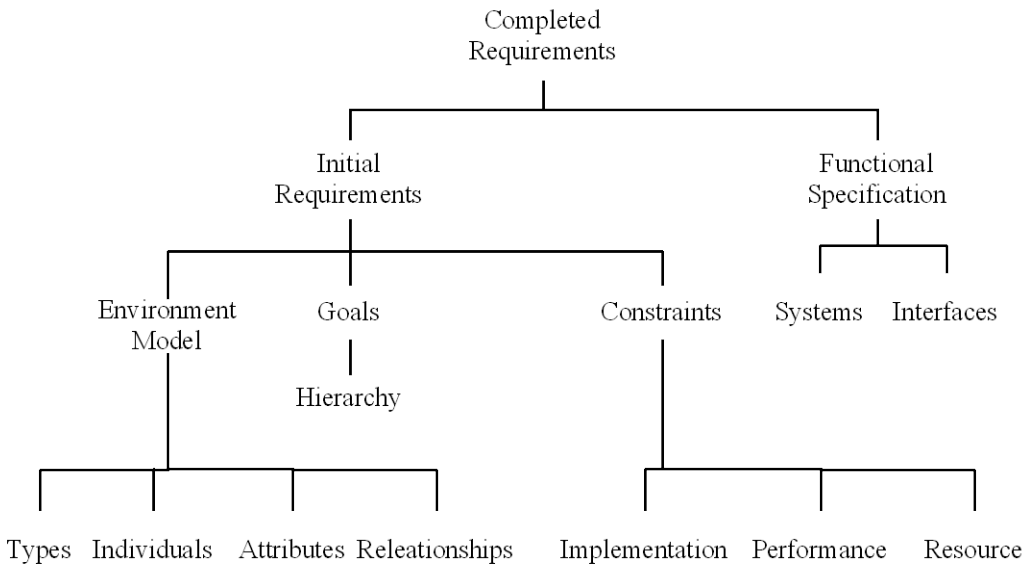

Figure 2. Structure of Requirements (From [5])

2. Derivation of Requirements

Figure 3 shows the flow from initial problem statement to RA to functional requirements to completed requirements. The outputs of RA (environment model, goal, and constraints) become the inputs to the functional specification. This research focuses on RA.

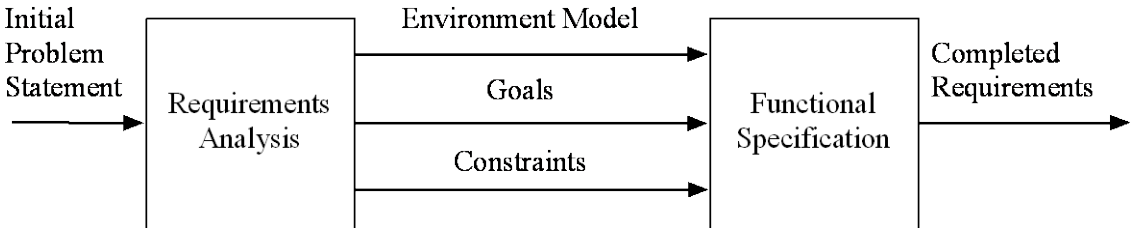

Figure 3. Derivation of Requirements (From [5])

3. Procedure for Requirements Analysis

The domain of software engineering is so vast that there are no universally accepted procedures for RA. As such, what follows is one approach to RA. Upon receiving the initial problem statement, it is analyzed to identify major concepts. These concepts are then used to build an environment model. High-level goals and constraints are captured and broken down into sub goals while continuously analyzing the problem. Looking at the problem from different perspectives will help to increase understanding of the problem and what it is the system must do. Figure 4 shows that as additional requirements and constraints are illuminated, the environment model is updated, and the process repeats itself. The analyst must be sure to inspect the goals looking for areas of the problem the customer has not considered. He should propose possible goals to the customer for consideration and include them upon concurrence. Nonconcurrence should be documented as a limitation on the scope of the project. Conferring with the customer must be a routine practice during RA to ensure buy-in, to receive feedback and any new ideas or requirements, and to ensure we had no diversion from the desired end state. When complete, a final version of requirements should be validated through customer reviews [5].

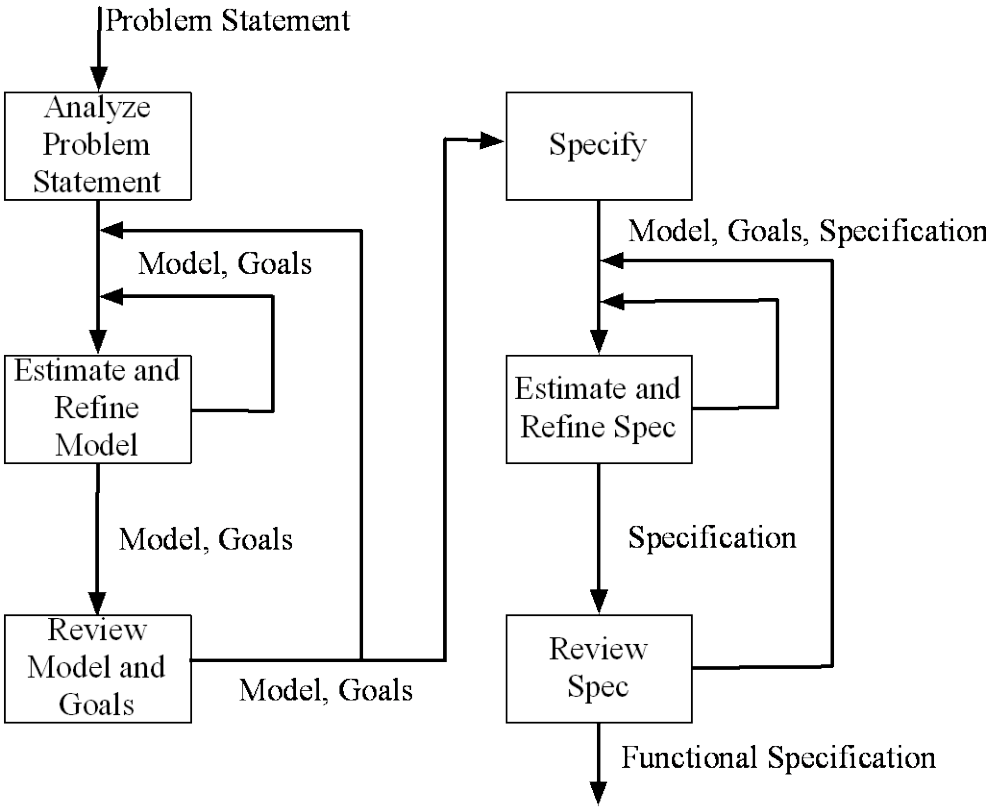

Figure 4. Procedure for Requirements Analysis (From [5])

With this high-level conceptual view of requirements analysis, we will now look at a few components that make up the foundation on which the completed requirements and software engineering process stand. These components are the initial problem statement, environment model, goal development, goal hierarchy, constraints, and interfaces.

B. INITIAL PROBLEM STATEMENT

The initial problem statement is a simple, high-level statement about the problem that needs to be solved. Because it is rare that the customer will be able to articulate the problem formally with sufficient detail, it is written, informally, in English from the customer's perspective. The initial problem statement is imprecise, short, and does not address many pertinent features of the problem. The analyst's job is to then develop a complete statement of the requirements [5].

C. ENVIRONMENT MODEL

Building the environment model is not an exact science and is a multistage process. It requires the skill of an aware, bright analyst to make educated guesses based on empirical knowledge. To start, a basic model is extracted from the initial problem statement and formalized by identifying the nouns it contains and the relationships between them. It is important to pick out the nouns that represent types of objects that will be present in the system. These objects and relationships can be written down in an outline format to ensure each one in the initial problem is called out. Observed properties are annotated and then verified by asking the customer to review the progress. A diagram can be generated to help visualize the model and will aid the analyst in communicating the model to others [5]. This process of guessing and then checking is important because the customer does not fully understand the problem and cannot precisely describe the need [6]. If the customer finds a discrepancy between the model and the intention, it is necessary to report it to the analyst so the model can be adjusted appropriately. Even after a model has progressed through many iterations of verification, it is never viewed as concrete as further modifications may be necessary in the future [5].

D. GOAL DEVELOPMENT

When developing goals, often times the customer is not clear on exactly what is needed because the breadth of the problem is not fully understood. The analyst must be careful to not assume that the customer has already documented and thought through the goals. Care must be taken to identify, elaborate, refine, and organize goals [7]. Running through different scenarios will be helpful to uncover and elaborate requirements and to answer questions about the system that would otherwise be difficult to answer [7]. In this section, we will use the terms agent and stakeholder instead of customer to more precisely discuss their involvement in goal development.

1. Agents and Stakeholders

"Agents are the entities or processes that seek to achieve goals within an organization or system based on the implicit responsibility that they must assume for the

achievement of certain goals" [7]. We identify agents so that we can determine which agent is responsible for the accomplishment of each goal [7].

Stakeholders are agents, but not all agents claim a stake in each goal. Only one agent is responsible for accomplishment, but there can be many stakeholders for a single goal. Each of these stakeholders might come from a different occupation or organization and might have a different viewpoint from other stakeholders [7].

When the project is used by an organization, there are many agents and stakeholders that need to be consulted to ensure all essential criteria have been considered. During requirements analysis, being able to represent the different criteria and distinguish between them will be an important skill [7].

The goals that are developed justify the need for the software system, but how are they developed and what happens if they change? To answer these questions, we will consider goal analysis and goal evolution.

Goal analysis is the process of identifying information about the organization that will help with recognizing, organizing, and classifying goals. Reviewing the organization's documentation is a good starting point. When reviewing the many types of documentation, using techniques like scenario analysis and identification of goal obstacles and constraints will help to analyze, elaborate, and refine the organization's goals [7].

Analysts hope that goals and requirements will remain the same from start to finish, but that rarely happens. As the agents and stakeholders learn more about what they need and what current technology can do, their requirements will constantly change. While this is true, often times a misunderstanding or misinterpretation of the requirements will be the cause for another round of requirements or goal refinement [8]. Let us consider goal analysis as a method to augment goal development, which is based solely on direct input from stakeholders.

2. Process

Goal analysis identifies goals through the review of the organization's literature and documentation. Identified goals are then organized and classified to aid in clarity and understanding. Goal evolution has to do with the volatility of goals from their identification to the system's implementation and throughout system maintenance [7].

Goals can be developed from various kinds of organizational information such as mission statements, rank structure, or job hierarchy structure. Also, transcripts of interviews with stakeholders can be analyzed to identify goals that were not explicitly communicated by the stakeholder. Stakeholders will often describe their requirements by using operational terms they are familiar with rather than expressing them as goals. Thus, picking out the verbs from their speech can be an effective way to derive the system goals [7].

We have been talking about agents and stakeholders so far but have not mentioned a great deal about them. For each goal that is identified, [7] states that the agents, stakeholders, and constraints must be identified [7]. Identifying agents should be done as soon as possible. This is done by understanding whom is ultimately responsible for reaching a stated goal. Identifying constraints furthers goal development because additional information is learned that reveals a requirement that should be included as a new goal. Look for words like *before, during,* or, *after* and for dependency relations to illuminate system contraints [7].

Goals evolve and change because the stakeholders' understanding improves; they reprioritize their requirements and realize behavior ramifications they might not be comfortable with implementing. These changes occur in the forms of elaboration and refinement [7].

The techniques of identifying obstacles, analyzing scenarios and constraints, and "operationalizing goals" [7] are useful for goal elaboration. Identifying obstacles helps to determine potential areas where the goals will fail so that those obstacles can be removed.

Considering different scenarios enables the analyst and stakeholders to envision the many possible cases that may arise from use of the system [7]. These scarios are commonly laid out in the form of use cases.

As goals are itemized and classified into groups of like goals or into a goal hierarchy, similar goals should be refined to eliminate redundancy and overlap. Two goals may be merged into one or a complex goal may need to be split into two or more goals [7].

3. Summary

We have discussed practical issues and the processes of goal analysis and goal evolution. Remember these key points from [7] when developing goals:

- More sources produce a more complete list of goals.
- Stakeholders provide insight and occupy unique vantage points from which they look at the project. Find them and use them.
- Categorizing goals may help to expose redundancies or overlap.
- System constraints may illuminate new goals.
- Scenarios help to develop a complete set of goals.

E. GOAL HIERARCHY

Communicating a system's goals can be clearly done through the use of a goal hierarchy. High-level goals are derived from the initial problem statement and are very informal. Each high-level goal expresses a goal of the customer that the system is expected to meet. In large projects, the initial problem statement may not be sufficient to provide the analyst all the high-level goals needed. Supplementary high-level goals may be added as the analyst and the customer develop a better understanding of the problem. A high-level goal is broken down into lower-level goals that specify how the system must operate to achieve that specific high-level goal. Second-level goals may be broken down

to a third and fourth-level goal, or further, if needed. Goals become more precise and narrower in focus on lower levels of the goal hierarchy [5]. A generic diagram of a goal hierarchy is provided in Figure 5.

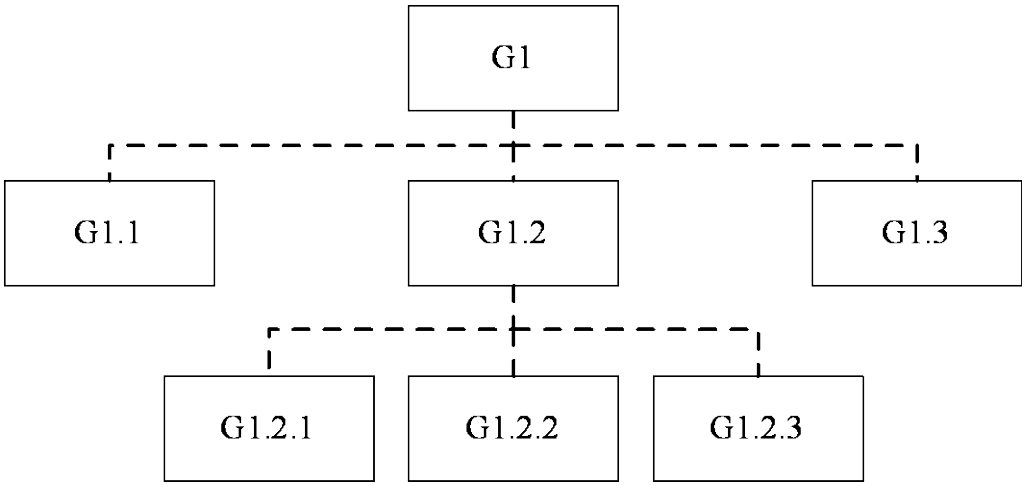

Figure 5. Goal Hierarchy (From [5])

F. CONSTRAINTS

Constraints seem like a burden to the analyst, but they also narrow the focus of the project. When embraced by the analyst, this narrowing of focus may serve to stimulate a fresh look at the problem [9]. There are three kinds of constraints typically associated with RA: implementation constraints, performance constraints, and resource constraints. Figure 6 illustrates these constraints in a diagram. We will discuss each of them briefly.

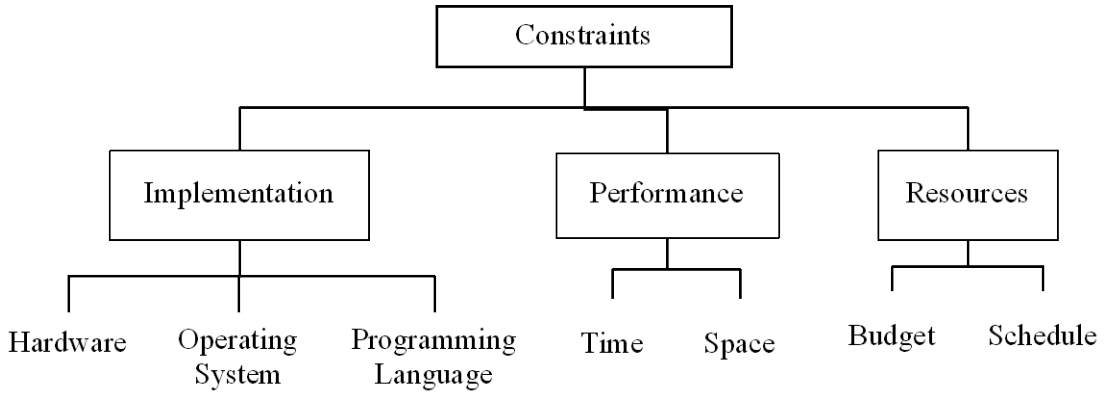

Figure 6. Constraints (From [5])

Large systems usually are restricted by implementation constraints in the form of implementation languages, pre-specified hardware configurations, operating systems, and predefined interfaces to existing systems. These constraints are usually provided by the customer and are easily specified [5].

Performance constraints relate to functions the system must perform and may be apparent in timing and space limitations. Such constraints would place a limit on memory consumption, system down time, system response time, error rate, or transaction frequency [5].

Resource constraints are imposed because of budget and schedule limitations. It is expected that system requirements be developed within budget and schedule constraints. As a problem grows in scope and complexity, the analyst may need to return to the customer and ask for resource constraints to be relaxed [5].

G. SUMMARY

This chapter introduced some of the relevant building blocks useful for conducting requirements analysis. The need for requirements analysis was explained, as were some general concepts to illustrate structure, derivation, and procedures. Several components were defined and illustrated with diagrams and goal development was

described in greater detail. This chapter was not a comprehensive survey of all aspects to consider during RA, but it should provide the reader with a basic understanding of the foundation on which Chapter III rests.

THIS PAGE INTENTIONALLY LEFT BLANK

III. REGISTRAR ADMINISTRATION SYSTEM (RAS) REQUIREMENTS ANALYSIS

We will consistently use the term *customer* to refer to the employees at MCU that participated in the review of all requirements analysis documents that were developed throughout the progress of this research. (We will refer to the customer in the masculine sense.) Primarily, the customer is made up of MCU employees who work in the Registrar's office and the Information and Education Technology Directorate.

A. INITIAL PROBLEM STATEMENT

We start out by determining the problem to be solved. The problem statement is expressed in the form of English narrative. There is no need for it to be expressed in technical language with which the customer may not be familiar. After a few rounds of discussion with the customer, we agreed on an initial problem statement. It is as follows:

> The purpose of this RAS is to automate recording and maintenance of student data, such as course completion and student grades, and to generate official transcripts and reports. The faculty member must be able to see all students in the course and make grade assignments. The Registrar must be able to see all courses taken by all students and their associated goals.

B. INITIAL ENVIRONMENT MODEL

As we begin to build the environment model, we identify the nouns in the problem statement. These nouns will become the foundation objects or types on which will be build the rest of the system. They are identified here in bold:

The purpose of this **RAS** is to automate recording and maintenance of student data, such as **course completion** and student **grades**, and to generate official **transcripts** and **reports**. The **faculty member** must be able to see all **students** in the course and make grade assignments. The **Registrar** must be able to see all courses taken by all students and their associated goals.

Identified users are faculty member, student, and Registrar. The objects are RAS, the notion of course completion, grade, transcript, and report. From these first steps, we can build our initial diagram of the environment model.

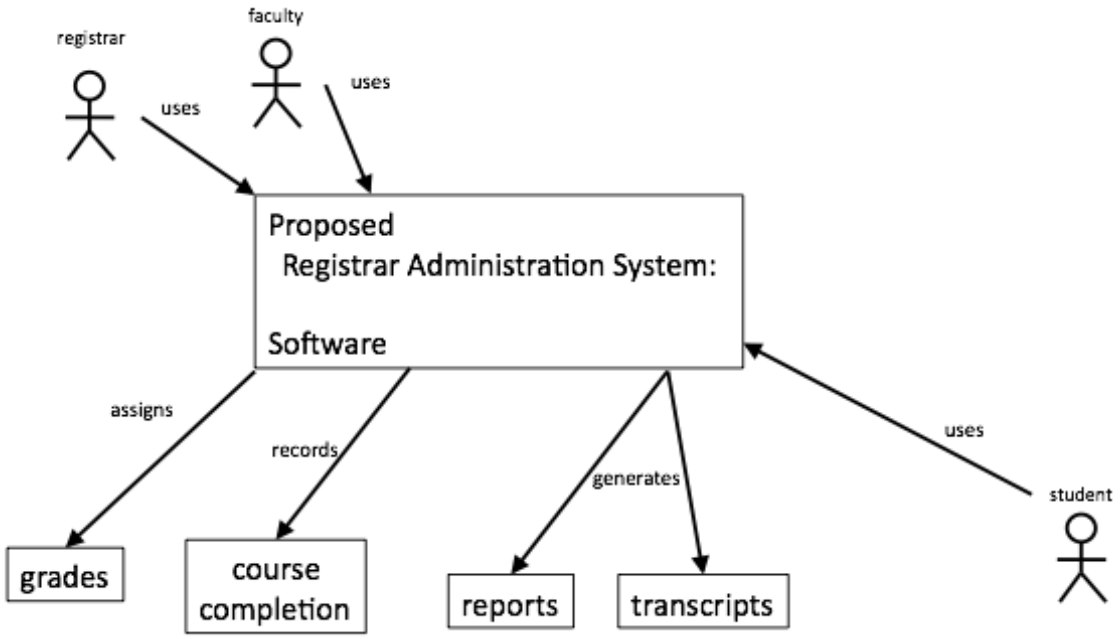

Figure 7. Initial Environment Model (After [5])

This model illustrates that the users (Registrar, faculty, student) use the system, RAS. RAS is used to assign grades. RAS is used to record course completion. RAS is used to generate official reports and transcripts.

C. INITIAL GOALS

1. High-Level Goals

Branching off from our initial problem statement, we identified the high-level goals we expect RAS to achieve. Since we are in the early stages of requirements analysis, these goals appear to be informally stated and may not be directly testable. These high-level goals will be decomposed into sub-goals, the achievement of which works together to achieve the high-level goal. This is true about each level of goals in

our goal hierarchy. We start by breaking our initial problem statement down into pieces and assign each one as a high level goal. We added a fourth goal to describe our desire for RAS to be robust.

G1: The purpose of this RAS is to automate recording and maintenance of student data, such as course completion and student grades.

G2: A second purpose of RAS is to automate generation of official transcripts.

G3: A third purpose of RAS is to automate generation of official reports.

2. Subgoals

High-level goals break down into subgoals to identify the component elements that must be met in order for the high-level goal to be met.

G1: The purpose of this RAS is to automate recording and maintenance of student data, such as course completion and grades.

- **G1.1:** The system must help the faculty member see all students in his or her course.
- **G1.2:** The system must let the faculty member make grade assignments.
- **G1.3:** The system must let the faculty member mark that student has accomplished all course requirements.

G2: A second purpose of RAS is to automate generation of official transcripts.

- **G2.1:** The system must let the Registrar see all of the courses.
- **G2.2:** The system must let the Registrar see all of the students.
- **G2.3:** The system must let the Registrar see all of the grades.
- **G2.4:** The system must bring all associated information for a given student together to allow printing of transcripts.

G3: A third purpose of RAS is to automate generation of official reports.

- **G3.1:** The system must allow the Registrar access to all statistics generated from course and student data.

D. INITIAL CONSTRAINTS

As we developed these goals, we identified associated constraints that must accompany implementation of that goal in order for RAS to function as intended.

C1: RAS contains only faculty and student data.

C1.1: Other employee data is maintained elsewhere.

C2: RAS must implement a robust data storage solution to house mission essential data to ensure data availability, security, integrity, and redundancy (c.f. G1).

C3: Official transcripts and reports must meet the Department of Defense's (DoD) requirements for protecting an individual's PII (c.f. G2, G3).

E. USE CASES

Now that we have listed our initial goals and constraints, we worked through the potential ways that RAS would be used and the likely users. We thought that four specific types of people or users would use RAS. They are the Registrar, faculty members, executive leadership, and students. One will notice that the user, executive leadership, was not mentioned in the initial problem statement or the initial environmental model. As we developed the goals and constraints, we discovered that we had unintentionally omitted MCU's civilian and military executive leadership that would require visibility of the data in the system and added it as another user that will access the system.

Next, we identified the functional uses for RAS. We knew that RAS needed to have access control, so we identified login as a fundamental use case. Other fundamental operations RAS would be expected to automate are assignment of grades, generation of official transcripts, generation of official reports, and the updating of personal data. This is not an exhaustive list due to time and resource constraints but these are some of the major use cases.

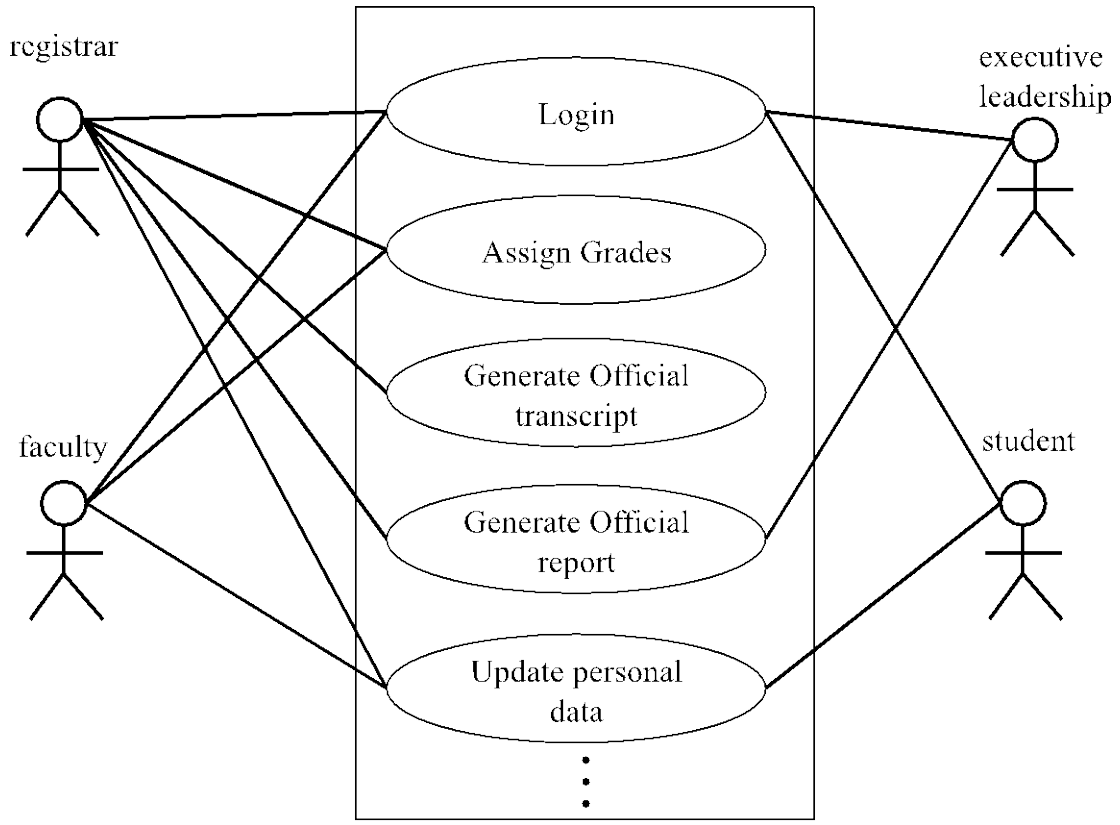

Figure 8. Initial Use Case Diagram

F. DESIGN NOTATION

To better grasp the large-scale design of RAS, we need to describe how the subsystems cooperate to achieve the goals of the overall system. We used our use cases as input into this stage of requirements analysis and drew a system level diagram. Next, we diagrammed each of the subsystems to better understand how their components work together to make the subsystem function. In the diagrams below, the large circle represents the system or subsystem of interest. Solid lines indicate internal component communication initiated in the direction of the arrowhead. Dashed lines indicate external system communication across an interface initiated in the direction of the arrowhead. It is understood that there will be two-way communication in the system. The labels on each of the lines are informative to describe the nature of the communication. The small bubbles inside the large circle depict the components that make up the large-scale system

or subsystem. The rectangles outside of the large circle represent objects that are external to the system or subsystem. These are objects for which an interface must be defined during a later phase of software engineering and is not part of this study.

1. Registrar Administration System as a System

RAS, at the system level, is composed of five subsystems: login, assign grades, generate official transcripts, generate official report, and update personal data. This diagram shows a subset of the external objects with which the subsystems must interface. Once a user logs in and is authenticated, he will be able to accomplish various tasks depending on his privilege level. He will accomplish those tasks by interacting with RAS through some method of user input/output (I/O).

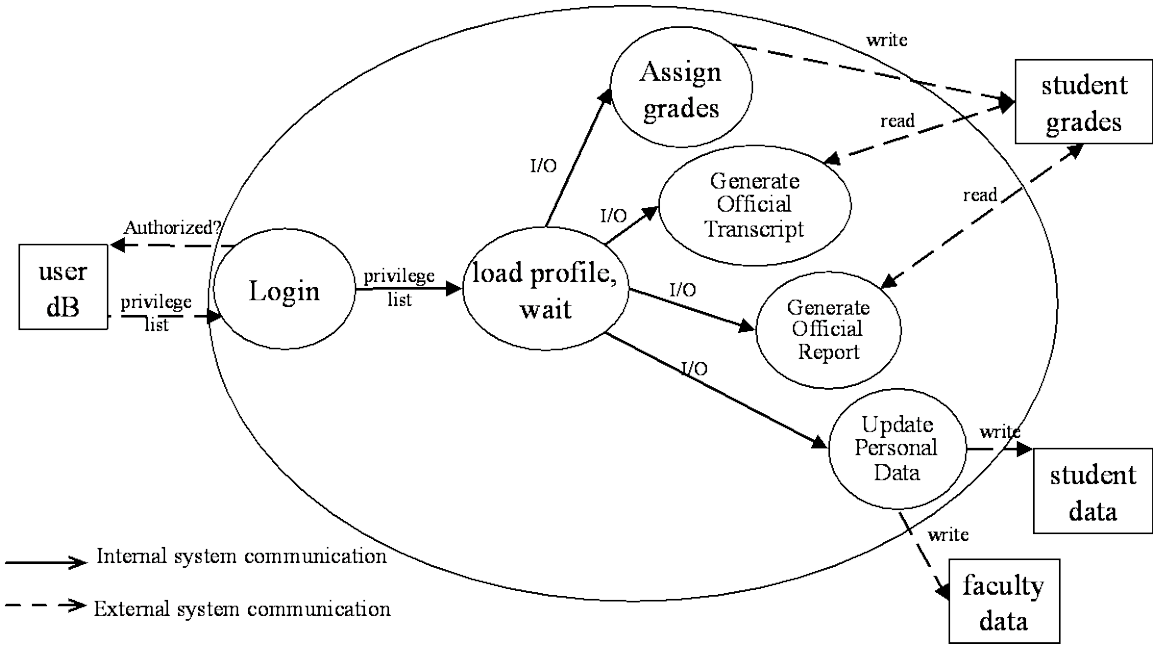

Figure 9. Registrar Administration System as a System (After [10])

2. Registrar Administration System Subsystems

For each subsystem listed, the users that are able to participate in the task are depicted in the upper left-hand corner of the diagram.

a. Login Subsystem

All authorized users are able to login to the system. A User database (dB) will be consulted to authenticate the credentials entered during login I/O. Once authenticated, that user's profile will load and receives a data feed depending on the type of user and his privilege level. He then is free to select a task to accomplish or to exit the system.

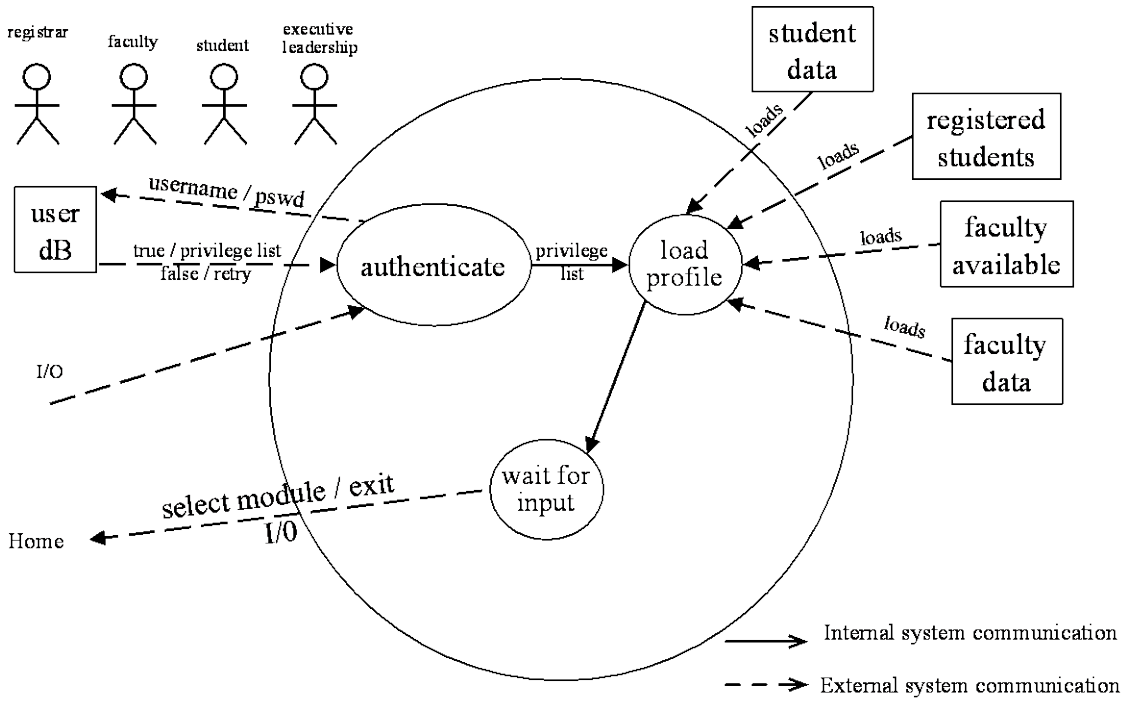

Figure 10. Login Subsystem (After [10])

b. Assign Grade Subsystem

The Registrar and faculty members are the only users able to make grade assignments. User I/O will bring the user into the assign grade subsystem upon login. Feeds coming from the registered students and course offerings databases combine to generate a list of students after a particular course is selected by the faculty member in preparation for grade assignment. Feeds coming from course offerings and faculty available databases combine to present the faculty member with a list of courses he is

currently teaching. Upon selection of the desired course for making grade assignments, the faculty member will be presented with the roster of students enrolled in that course. After grade assignment is complete, the faculty member should save those grades to the student grades database. This database contains the grade assignments. The user can then observe the assigned grades and exit the subsystem.

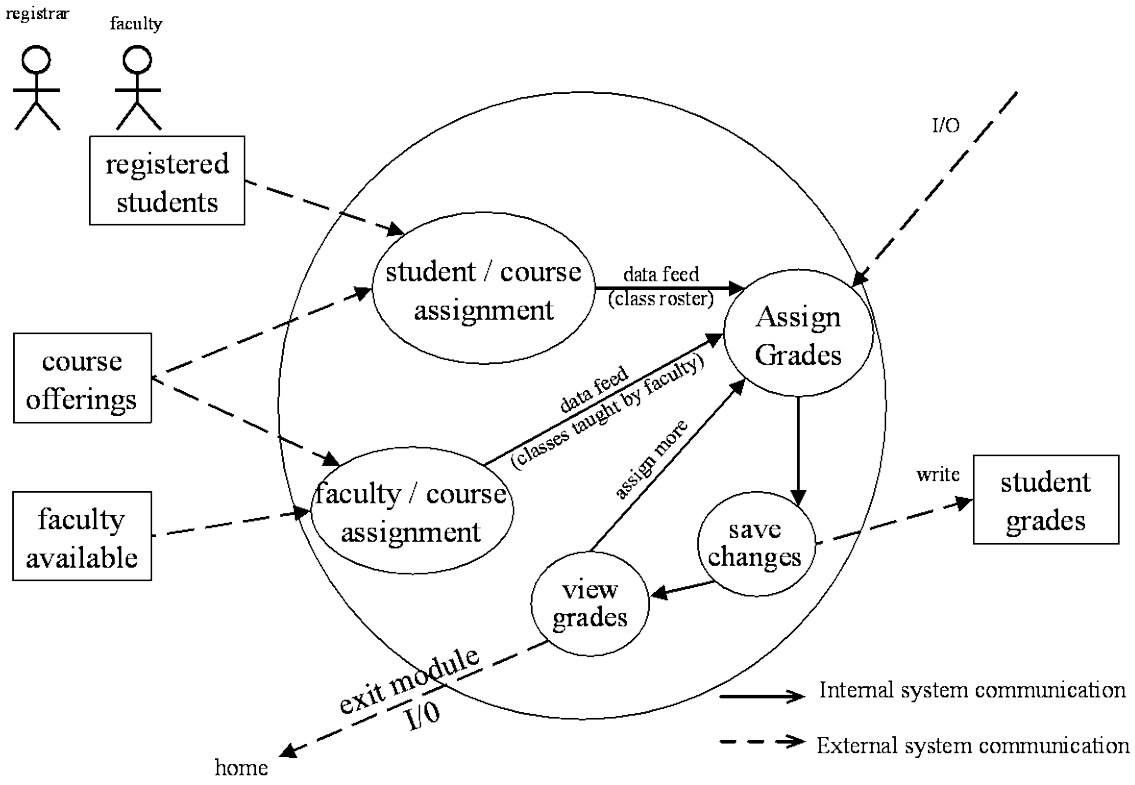

Figure 11. Assign Grade Subsystem (After [10])

c. Generate Official Transcript Subsystem

The Registrar is the only user that has access to the Generate Official Transcript subsystem. User I/O will bring the Registrar into the Generate Official Transcript subsystem upon login. A feed coming in from the registered students database will allow the Registrar to choose a student or batch of students for whom to generate a transcript. Once selected, the student profile data, courses taken and associated grades will be loaded via the interfaces that link RAS to the student data and student grades

databases, respectively. The Registrar will now be able to view the transcript(s), which he can print on the appropriate letterhead for an official version. He can email or save an unofficial version to disk. The register can then exit the subsystem or loop back to generate additional transcripts.

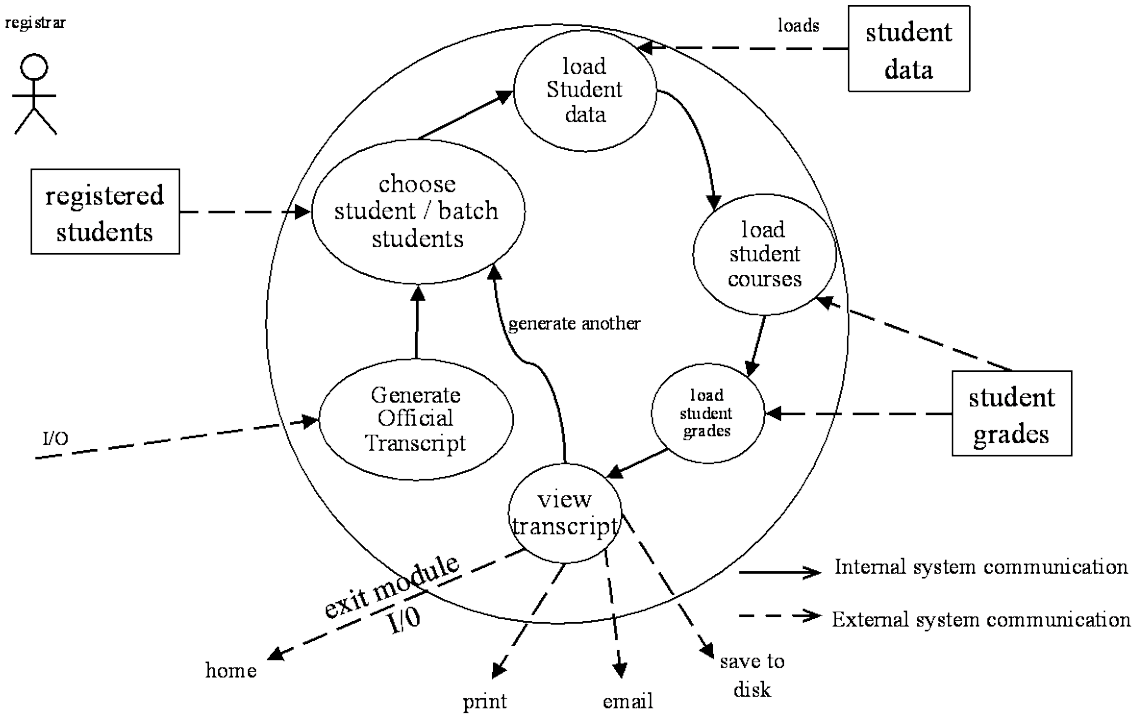

Figure 12. Generate Official Transcript Subsystem (After [10])

d. Generate Official Report Subsystem

The Registrar and executive leadership are the users that have access to the Generate Official Report subsystem. User I/O will bring the user into the Generate Official Report subsystem upon login. There are a myriad of reports that RAS must allow the user to build and customize. Developing the functionality to build customizable reports is beyond the scope of this research and is an area for future work. Feeds coming in from the course offerings and registered students database will allow the user to choose parameters to include in the report. Once selected, the appropriate data will be loaded via the interfaces that link RAS to the registered students and student

grades databases, respectively. The user will now be able to view the report, which he can print, email, or save to disk. The user can then exit the subsystem or loop back to generate additional reports.

For the sake of simplicity, Figure 13 depicts a specific scenario for which a report may need to be generated. The scenario is that students have been taking a particular course for the past seven years and the Commanding General of TECOM wants to know which students have completed that course in the past five years and what their grades were.

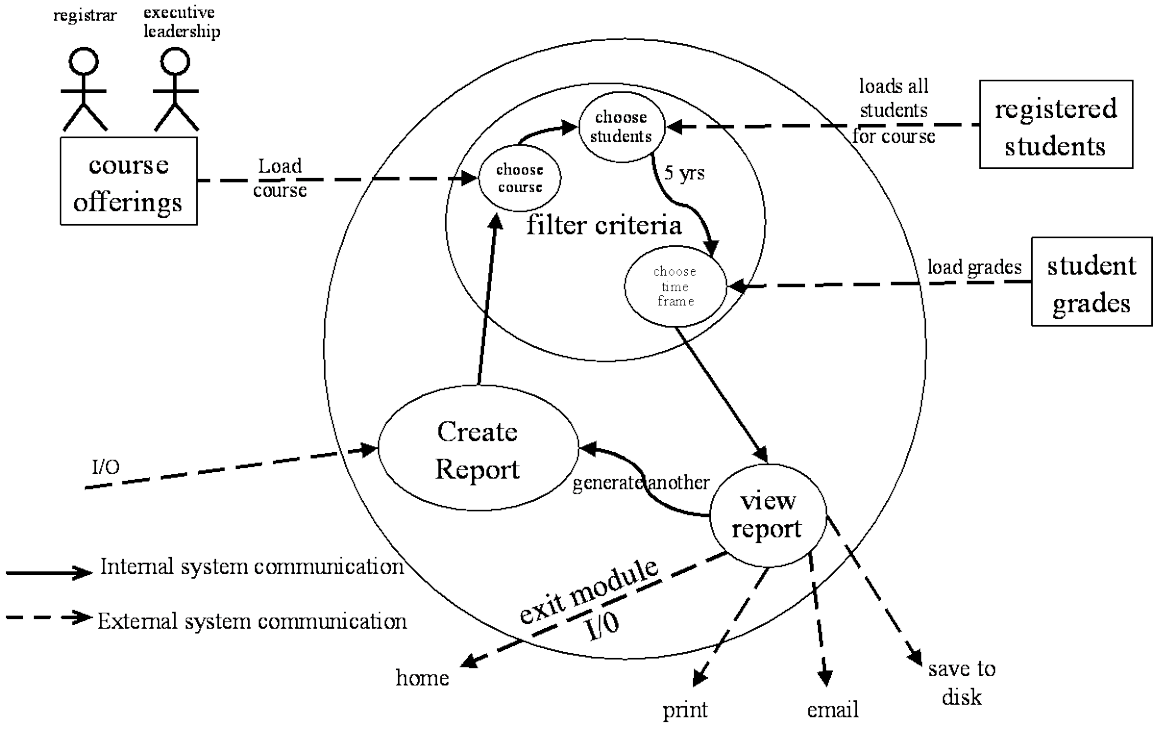

Figure 13. Generate Official Report Subsystem (After [10])

e. Update Personal Data Subsystem

The Registrar, faculty members, and students are the users that have access to the Update Personal Data subsystem. User I/O will bring the user into the subsystem upon login. Once logged in, that user's profile data will be loaded from the

appropriate faculty or student data database via the external interfaces. From here, the user will be able to view his personal data. When the option to edit personal data is selected the user can enter the desired information and then save it. The action to save changes will write the updated data to the database. The user can then observe his updated personal data, print it, email it, save it to disk, or make additional modifications. When ready, the user can exit the subsystem.

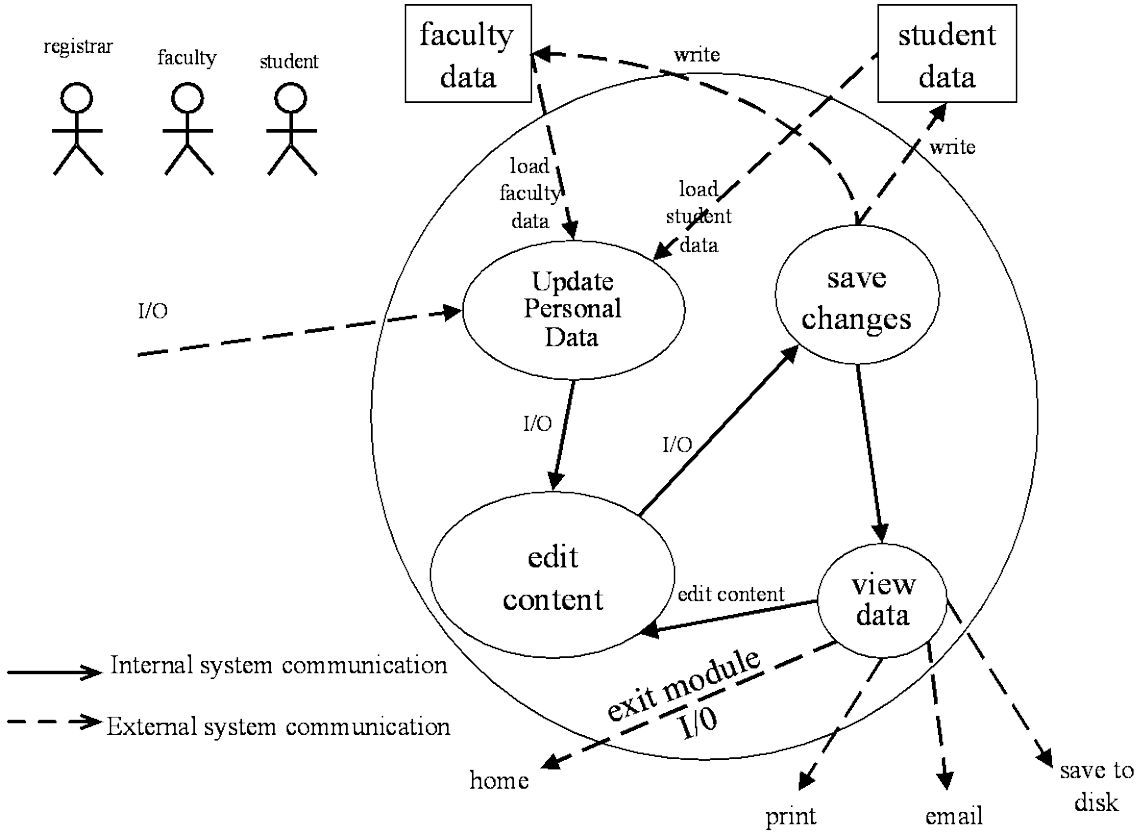

Figure 14. Update Personal Data Subsystem (After [10])

G. INTERFACES

After working through the design notation, we have developed a better understanding of what we expect the system, RAS, and its subsystems to do. This has helped illuminate some of the external entities or objects that might need to be created or accessed to meet RAS's information requirements. In Figure 15, the external objects are

depicted as rectangles. The system, RAS, is depicted by the innermost oval. The interfaces to the objects are depicted as wedges in the oval band that surrounds the RAS internal system. Spokes, representing communication links, connect the external objects to RAS via the appropriate interfaces.

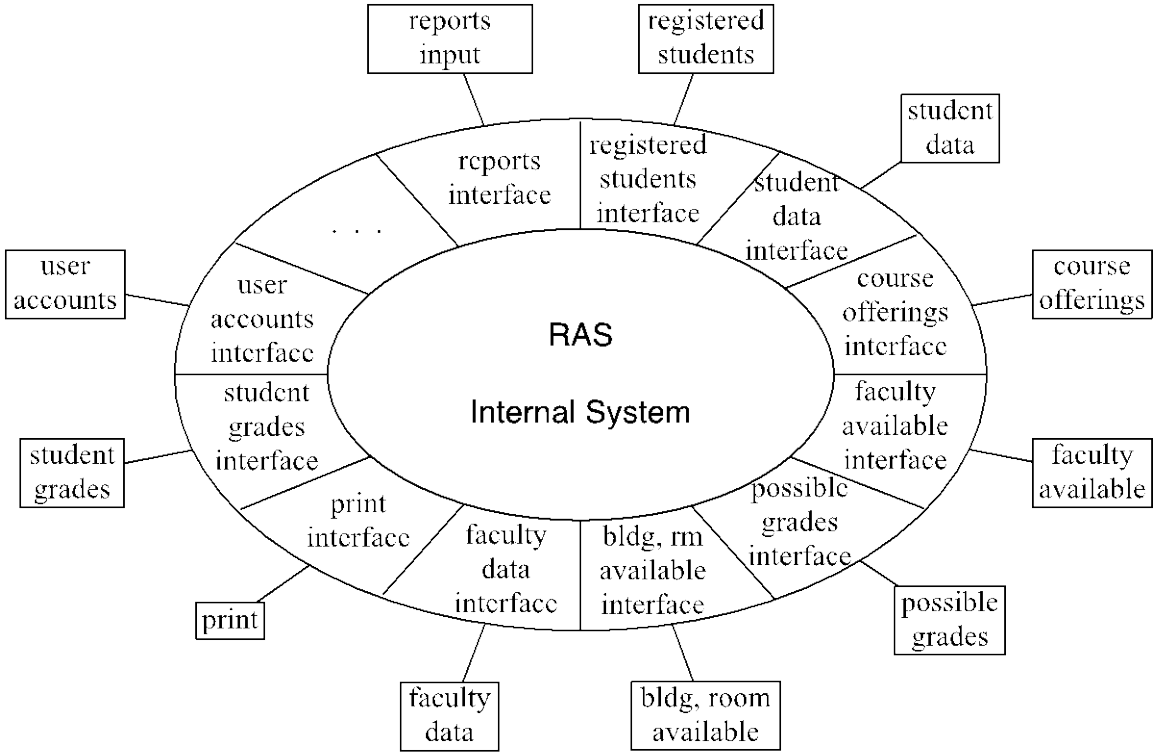

Figure 15. External Interfaces (After [5])

H. CONTEXT DIAGRAMS

Now that we have identified users of the system, observed RAS from the use case perspective, and identified some external interfaces, we will now observe RAS from the user's perspective in the form of context diagrams to draw some additional insight into our problem. The process of developing these context diagrams forced us to think of the system in a different way than we had before. We added several additional goals, sub-goals, and constraints as a result. These are described in the Goal Refinement section.

1. Student

The student's interaction with the system allows him to view and modify his own student data and view his own grades. To provide service to a student user, RAS must be able to print and interface with the student data and student grades databases.

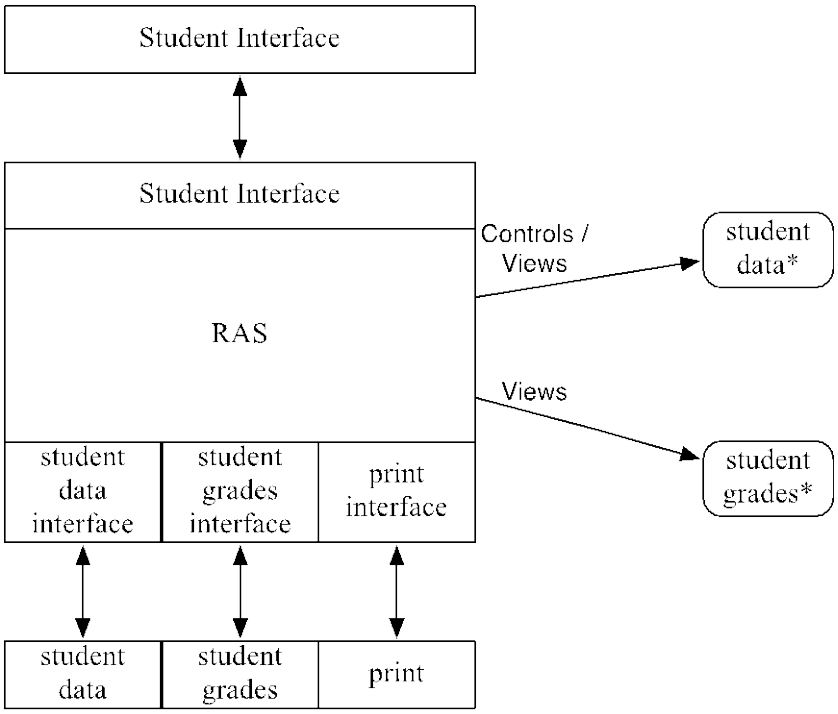

Figure 16. Student Context Diagram (After [5])

2. Faculty

The faculty members' interaction with the system allows him to view and modify his own faculty data and the students' grades for the courses he is teaching. He also must be able to view the student data of the students taking his courses and the range of possible grades when he is using the Assign Grades subsystem. To provide service to a faculty user, RAS must be able to print and interface with the student data, student grades, faculty data, and possible grades databases.

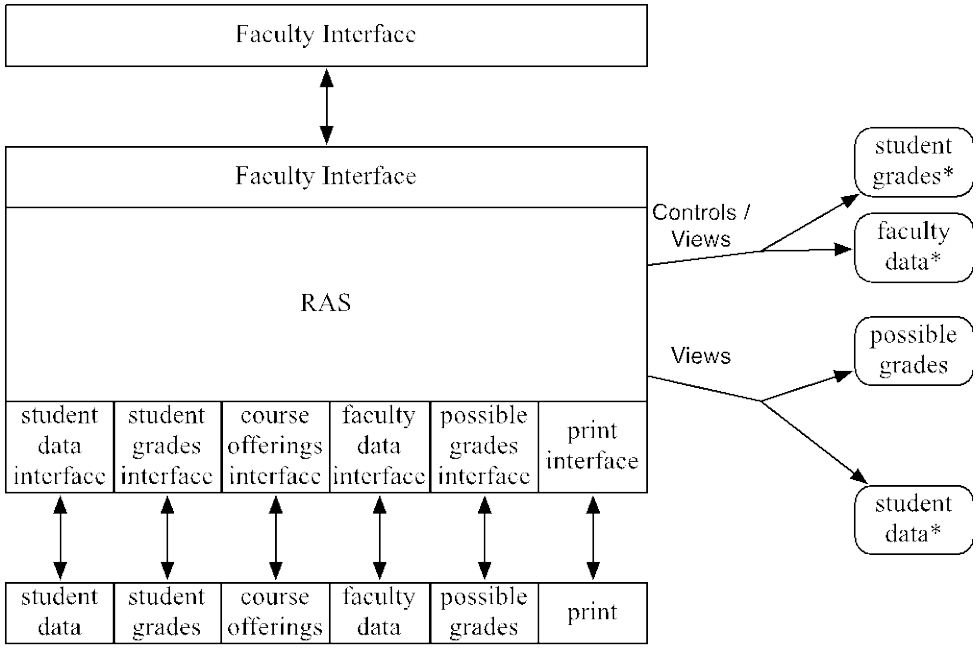

Figure 17. Faculty Context Diagram (After [5])

3. Registrar

The Registrar's range of interaction allows him the highest level of access in the system. He must be able to view and modify user accounts, student grades, registered students, student data, faculty data, reports, and the possible grades available to faculty in the Assign Grades subsystem. To provide service to a Registrar user, RAS must be able to print, generate reports, and interface with the users accounts, registered students, student data, student grades, faculty data, and possible grades databases.

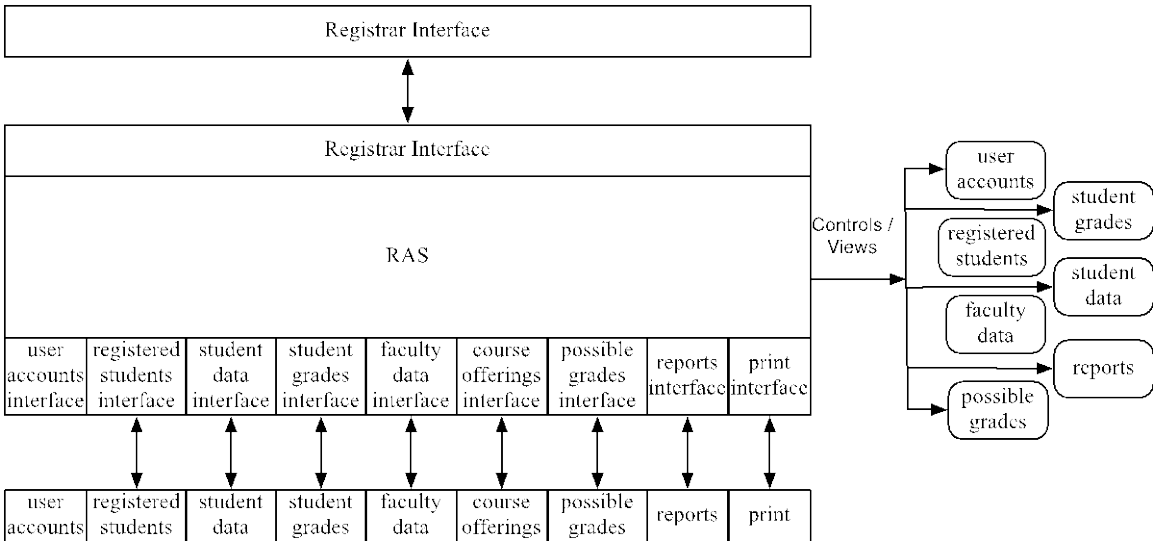

Figure 18. Registrar Context Diagram (After [5])

4. Executive Leadership

The executive leaderships' interaction with the system allows them to view registered students, student grades, student data, faculty data, and reports. To provide service to the executive leadership users, RAS must be able to print, generate reports, and interface with the registered students, student data, student grades, and faculty data databases.

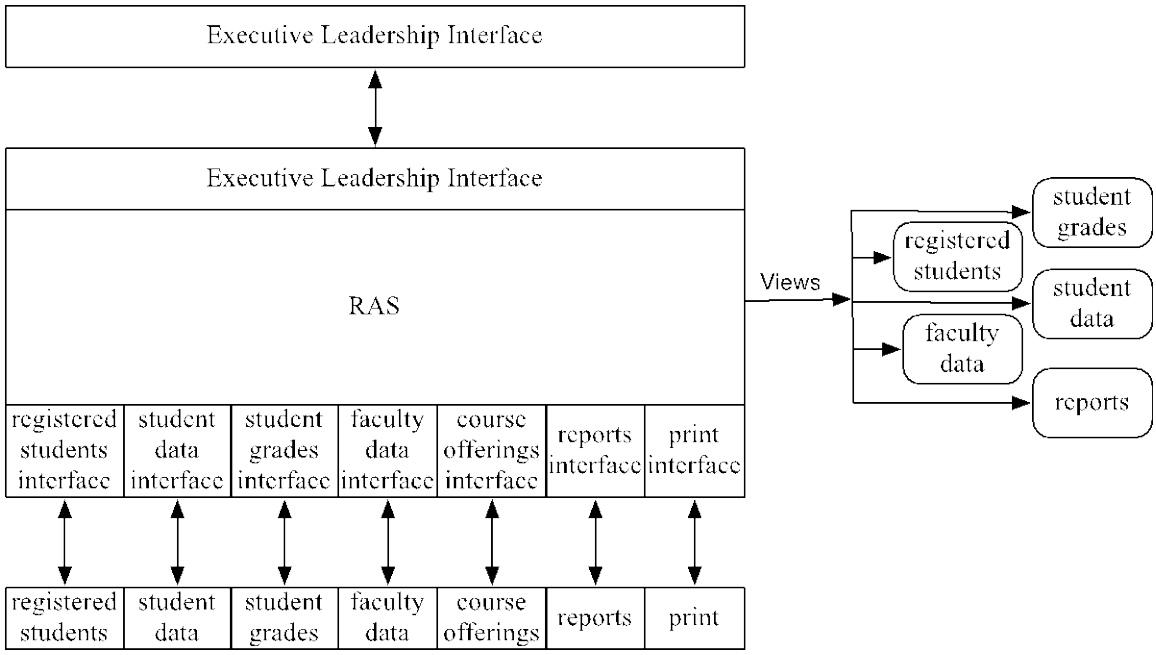

Figure 19. Executive Leadership Context Diagram (After [5])

I. USER SURVEY

To get a more complete idea about what this system should do we developed a survey and distributed it to a representative population of perspective users at MCU. The survey was sent to Registrar staff, faculty members, and executive leadership. We received eighteen responses to the survey: two were from Registrar staff, ten from faculty, and six from executive leadership.

The survey illuminated seven major take-a-ways or concepts, three of which were added as constraints, that the requirements analysis of RAS should include. The new constraints are marked with an asterisk (*) and are included in the Constraints Refinement paragraph below. The seven takeaways are: 1) that the survey confirmed a need of such a system at MCU for standardization and to reduce or eliminate errors when transferring grades 2)* that after a faculty member uploads grades, RAS business rules must be flexible to establish, update, and refine grade approval workflow to ensure the appropriate people can review the grades before they are final 3) that RAS should be implemented for all schools at MCU to standardize procedures 4)* that the system must

address security concerns to allow secure access to RAS from an un-trusted domain 5) that a mechanism is needed to allow variable grade submission deadlines for different courses across different schools 6)* that grades should never be frozen; a mechanism must be provided to correct inaccurate data 7) that the overall system must be flexible, not static, to handle the many unique situations found throughout MCU.

J. GOAL REFINEMENT

The customer reviewed our initial goals to maintain his buy-in, to receive feedback and any new ideas or requirements, and to ensure we had not diverted from his desired end state. We received additional insight into how his organization operates and expanded our initial list of goals. The additions and modifications are found here. To simplify representation of the results of the goal refinement process, we combine all goals learned during all iterations of goal refinement here in a single section. It is important for the reader to remember that goal refinement is not a one-time event. It is an iterative, continuous process conducted by the analyst throughout the requirements analysis process. A completed goal hierarchy will follow later in this study.

During goal refinement, an additional user, alumni, was identified that must be considered. The associated goals for alumni are found under G5.

G1: G1 was modified and a sub-goal for G1 was added. G1 now states, "The purpose of this RAS is to automate recording and maintenance of student data, such as course completion, student grades, and faculty data."

Goal G1.1 was expanded to include:

> **G1.1.1:** The system must associate faculty member with all courses taught by that faculty member.
>
> > **G1.1.1.1:** The faculty member must be presented with a listing of courses he has taught for selection and further action (i.e., further action as allowed by future MCU policy).
>
> **G1.1.2:** For a given course, the system must associate all students that have taken or are taking that course.

G1.1.2.1: For a given course, the system must use some mechanism to distinguish one course from another as a single course may be offered multiple times in the same year and over many years.

Goal G1.2 was expanded to include:

G1.2.1: The system must have the range of possible grades loaded and available to the faculty member during grade assignment action.

G1.2.2: The system must present the student names in the selected course to the faculty member and offer a mechanism to select a grade to assign.

G1.2.3: The system must provide a mechanism for the faculty member to upload the grade assignments to the central database.

Goal G1.3 was expanded to include:

G1.3.1: The system must associate the presence of a grade for a student in a course other than the grade of 'F', 'INC', or 'NM' with course completion. Absence of a grade or a grade of 'F', 'INC', or 'NM' must be associated as course incomplete. 'F'- fail. 'INC' – incomplete. 'NM' – non-mastery.

G1.4: Goal G1.4 was added, which states, "The system must assign a unique student identification number to each student record."

G1.5: Goal G1.5 was added, which states, "The system must provide a mechanism that allows students to view and update their own personal data over the Internet."

G1.6: Goal G1.6 was added, which states, "The system must provide a mechanism that allows students to view their own grades over the Internet."

G1.7: Goal G1.7 was added, which states, "The system must provide a mechanism that allows faculty to update their own personal data over the Internet."

G1.8: Goal G1.8 was added, which states, "The system must provide a mechanism that allows faculty to view, over the Internet, only the personal data of the students currently enrolled in a course that faculty member is teaching."

G2.4: Goal G2.4 was modified to state, "The system must bring all associated information for a given student together to allow generation of transcripts." *Printing* of transcripts was changed to read *generation* of transcripts.

G2.4.1: The system must provide a mechanism for the Registrar to generate a batch of transcripts vice only one at a time.

G2.5: Goal G2.5 was added, which states, "The system must allow the Registrar to print an official transcript."

G2.6: Goal G2.6 was added, which states, "The system must calculate the student's grade point average (GPA).

Goal G3 was expanded as follows:

G3.1: Goal G3.1 was modified to include Executive Leadership and now states "The system must allow the Registrar and Executive Leadership access to all statistics generated from course and student data."

G3.2: The system must allow the Registrar and Executive Leadership access to all student and faculty data.

The purpose of adding sub-goal G3.2 is to ensure the multiple pieces of demographic information about a student and faculty member are accessible for the generation of official reports.

G3.3: The system must allow the Registrar and Executive Leadership to build custom reports.

G3.3.1: Once a custom report is built, the system must allow that report to be saved for future reuse, modification, or deletion.

G4: Goal G4 was added, which states, "The system must provide a mechanism for Executive Leadership to view student and faculty data, student grades, and official reports."

G5: Goal G5 was added and expanded, which states, "RAS must automate transition of a student from the status of *student* to *alumni* upon graduation. This must include migration of that student's profile and record to the alumni sector if alumni data are to be stored separate from student data."

> **G5.1** The system must provide a mechanism that allows alumni to view and update their personal and contact information over the Internet.

Goal G5 and sub-goal G5.1 were added because MCU has three classes of information requirements for each student. These classes are 1) student is registered, but classes have not yet started 2) student is progressing through program of instruction 3) student graduates and becomes alumni.

G6: Goal G6 was added and expanded, which states, "The Registrar must be authorized to modify all personal and academic data in RAS to serve as the failsafe mechanism to recover from erroneous data or help users that need assistance."

G6.1: The system must interface with existing automated registration procedures and allow the Registrar to accept or import new student records into RAS. This would add a new student to the pool of registered students.

G6.1.1: The system must allow the Registrar to drop student records. This would allow for recovery after an error.

G6.2: The system must allow the Registrar to edit student grades. This would allow Registrar to assist faculty if needed (i.e., faculty no longer works for MCU).

G6.3: The system must allow the Registrar to edit student data

G6.4: The system must allow the Registrar to edit faculty data.

G7: Goal G7 was added, which states, "The system must provide a mechanism for logging of all system transactions with sufficient detail to determine what was done and by whom."

G8: Goal G8 was added, which states, "The system must be able to assign some user the role of super user that enables him to grant privileges to all other users."

G9: Goal G9 was added, which states, "The system must be flexibly designed to allow for future expansion or modification of capabilities."

G10: Goal G10 was added, which states, "The system must not require data entry that is already being done through existing USMC systems."

K. CONSTRAINTS REFINEMENT

As the use cases, design notation, interfaces, context diagrams, and refined goals were iteratively developed, additional constraints were illuminated that we had not before considered. We linked each constraint to the corresponding goal(s) to which it relates. Those goals are enclosed in parentheses following the constraint as a cross-reference. To simplify representation of the results of the constraints refinement process, we combine all constraints learned during all iterations of constraints refinement here in a single section. It is important for the reader to remember that constraints refinement is not a one-time event. It is an iterative, continuous process conducted by the analyst throughout the requirements analysis process. A completed constraints list will follow later in this study.

C4: Available grade choices to faculty for assignment to student must be a hard list to provide for standardization of entry across all faculty members (c.f. G1.2.1).

C5: The system must allow faculty members to only assign grads to their own students in a course the faculty member is actively teaching (c.f. G1.2.2).

C6: The system must not require the faculty member to assign grades to all students before grades can be uploaded to the database (c.f. G1.2.3).

C7: The system must not associate a student with a course until the student has registered and been approved for the course (c.f. G1.3.1).

C8: The system must provide a mechanism to protect student and faculty personally identifiable information (PII) (c.f. G1.4).

C9: A student can only view and update own personal data (c.f. G1.5).

C10: A student can only view own grades (c.f. G1.6)

C11: A faculty member can only view and update own personal data (c.f. G1.7).

C12: A faculty member can only see student data for students enrolled in his or her course (c.f. G1.8).

C13: Official transcripts can only be printed. They cannot be sent via email or saved to portable media. However, an unofficial transcript can be sent via email or be saved to portable media (c.f. G2.5).

C14: The system must allow a student's transition to *alumni* status only after all requirements for graduation have been met (c.f. G5).

C15: Alumni can only view and update own personal data (c.f. G5.1).

C16: The system must provide a mechanism to prevent an authorized user with Registrar permissions from maliciously or accidentally deleting student or alumni records (c.f. G6.1.1). (i.e., two-person integrity on certain transactions)

C17: The system must calculate GPA based on MCU's current procedures. (c.f. G2.6)

C18: The system must not allow Executive Leadership to modify data. (c.f. G4)

C19: The system must provide a mechanism to modify user permissions and granting of authority to privileged users. (c.f. G8)

C20: The system must not allow the super user and Registrar to have the same personally identifiable credentials. (c.f. G8)

C21: The system must interface with existing USMC systems that are actively in use to prevent duplicate data entry and duplicate MCU student registration efforts. (c.f. G10)

C22: After a faculty member uploads grades, the system's business rules must be flexible to establish, update, and refine grade approval workflow to ensure the appropriate people can review the grades before they are final. (c.f. G1)

C23: The system must address security concerns to allow secure access to RAS from an un-trusted domain. (c.f. G1.5, G1.6, G1.7, G1.8)

C24: The system must never allow grades to be frozen. A mechanism must be provided to correct inaccurate data. (c.f. G6.2)

L. ENVIRONMENT MODEL REFINEMENT

Refinement of our goals and constraints has led us to a better understanding of our software environment. The reader will notice that we have added two additional users and four additional objects to our environment model as seen in Figure 20. The new users are executive leadership and alumni. The reader will note that we have already incorporated executive leadership in our use case diagram in Figure 8. The new objects are student data, faculty data, alumni data, and GPA. Refining the goals and constraints with the customer helped us to identify the need for these added components to RAS. In this refined environment model, we have linked each object and users' interaction with RAS to the goal(s) that require that interaction. The goals are listed adjacent to the action words and are enclosed in parentheses.

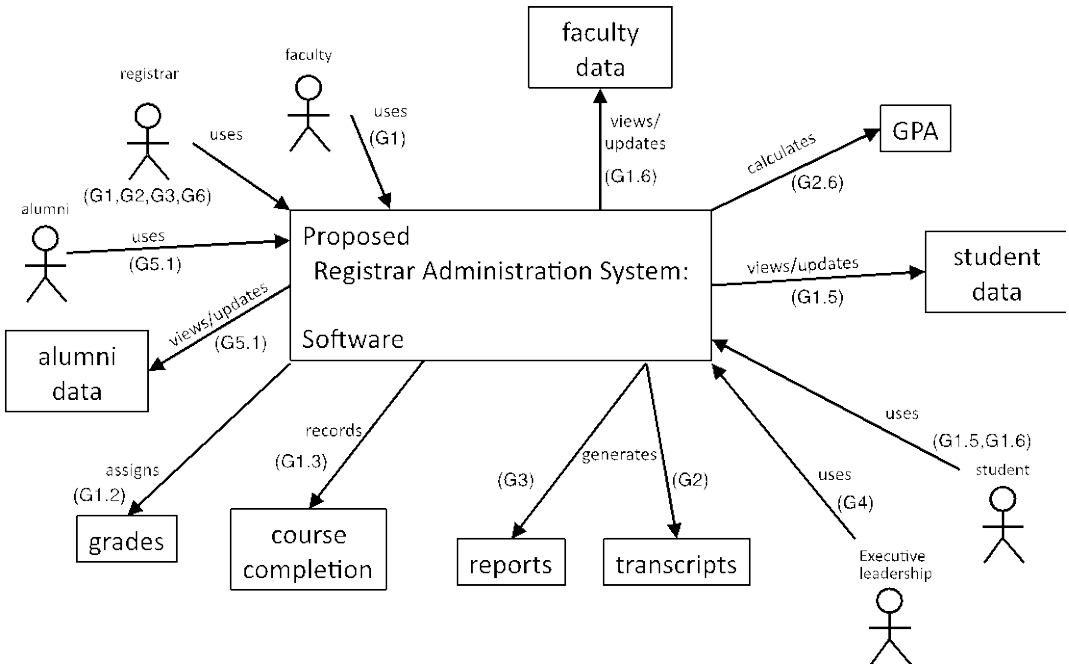

Figure 20. Refined Environment Model (After [5])

We also refined our use case diagram to include the new user we discovered, alumni.

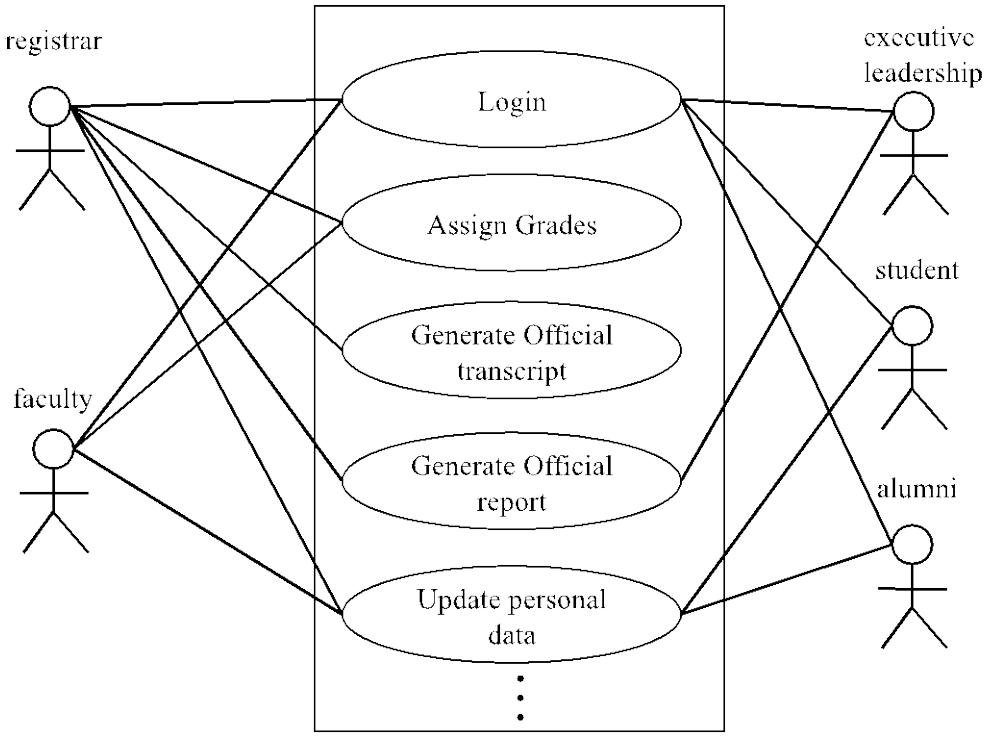

Figure 21. Refined Use Case Diagram

M. COMPLETED GOAL HIERARCHY

After many iterations of reviewing, considering, communicating, deciding, and refining with the customer, the completed goal hierarchy is listed here. The completed list of constraints is found in the next section.

G1: The purpose of this RAS is to automate recording and maintenance of student data, such as course completion, student grades, and faculty data."

G1.1: The system must help the faculty member see all students in their course.

G1.1.1: The system must associate faculty member with all courses taught by that faculty member.

G1.1.1.1: The faculty member must be presented with a listing of courses he or she has taught for selection and further action (i.e., further action as allowed by future MCU policy).

G1.1.2: For a given course, the system must associate all students that have taken or are taking that course.

G1.1.2.1: For a given course, the system must use some mechanism to distinguish one course from another as a single course may be offered multiple times in the same year and over many years.

G1.2: The system must let the faculty member make grade assignments.

G1.2.1: The system must have the range of possible grades loaded and available to the faculty member during grade assignment action.

G1.2.2: The system must present the student names in the selected course to the faculty member and offer a mechanism to select a grade to assign.

G1.2.3: The system must provide a mechanism for the faculty member to upload the grade assignments to the central database.

G1.3: The system must let the faculty member mark that student has accomplished all course requirements.

G1.3.1: The system must associate the presence of a grade for a student in a course other than the grade of 'F', 'INC', or 'NM' with course completion. Absence of a grade or a grade of 'F', 'INC', or 'NM' must be associated as course incomplete. 'F'- fail. 'INC' – incomplete. 'NM' – non-mastery.

G1.4: The system must assign a unique student identification number to each student record.

G1.5: The system must provide a mechanism that allows students to view and update their own personal data over the Internet.

G1.6: The system must provide a mechanism that allows students to view their own grades over the Internet.

G1.7: The system must provide a mechanism that allows faculty to update their own personal data over the Internet.

G1.8: The system must provide a mechanism that allows faculty to view, over the Internet, only the personal data of the students currently enrolled in a course that faculty member is teaching.

G2: A second purpose of RAS is to automate generation of official transcripts.

G2.1: The system must let the Registrar see all of the courses.

G2.2: The system must let the Registrar see all of the students.

G2.3: The system must let the Registrar see all of the grades.

G2.4: The system must bring all associated information for a given student together to allow generation of transcripts.

G2.4.1: The system must provide a mechanism for the Registrar to generate a batch of transcripts vice only one at a time.

G2.5: The system must allow the Registrar to print an official transcript.

G2.6: The system must calculate the student's GPA.

G3: A third purpose of RAS is to automate generation of official reports.

G3.1: The system must allow the Registrar and Executive Leadership access to all statistics generated from course and student data.

G3.2: The system must allow the Registrar and Executive Leadership access to all student and faculty data.

G3.3: The system must allow the Registrar and Executive Leadership to build custom reports.

G3.3.1: Once a custom report is built, the system must allow that report to be saved for future reuse, modification, or deletion.

G4: The system must provide a mechanism for Executive Leadership to view student and faculty data, student grades, and official reports.

G5: RAS must automate transition of a student from the status of *student* to *alumni* upon graduation. This must include migration of that student's profile and record to the alumni sector if alumni data are to be stored separate from student data.

G5.1 The system must provide a mechanism that allows alumni to view and update their personal and contact information over the Internet.

G6: Goal G6 was added and expanded which states, "The Registrar must be authorized to modify all personal and academic data in RAS to serve as the failsafe mechanism to recover from erroneous data or help users that need assistance."

G6.1: The system must interface with existing automated registration procedures and allow the Registrar to accept or import new student records into RAS. This would add a new student to the pool of registered students.

G6.1.1: The system must allow the Registrar to drop student records. This would allow for recovery after an error.

G6.2: The system must allow the Registrar to edit student grades. This would allow Registrar to assist faculty if needed (i.e., faculty no longer works for MCU).

G6.3: The system must allow the Registrar to edit student data

G6.4: The system must allow the Registrar to edit faculty data.

G7: The system must provide a mechanism for logging of all system transactions with sufficient detail to determine what was done and by whom.

G8: The system must be able to assign some user the role of super user that enables him to grant privileges to all other users.

G9: The system must be flexibly designed to allow for future expansion or modification of capabilities.

G10: The system must not require data entry that is already being done through existing USMC systems.

N. COMPLETED CONSTRAINTS

RAS's constraints are listed here to itemize the complete listing of constraints developed during the Requirements Analysis phase. Each constraint has been classified as an implementation constraint or performance constraint. Currently, there are no resource constraints.

1. Performance Constraints

C1: RAS contains only faculty and student data.

C1.1: Other employee data is maintained elsewhere.

C2: RAS must implement a robust data storage solution to house mission essential data to ensure data availability, security, integrity, and redundancy (c.f. G1).

C3: Official transcripts and reports must meet DoD's requirements for protecting an individual's PII (c.f. G2, G3).

C4: Available grade choices to faculty for assignment to student must be a hard list to provide for standardization of entry across all faculty members (c.f. G1.2.1).

C5: The system must allow faculty members to only assign grads to their own students in a course the faculty member is actively teaching (c.f. G1.2.2).

C6: The system must not require the faculty member to assign grades to all students before grades can be uploaded to the database (c.f. G1.2.3).

C7: The system must not associate a student with a course until the student has registered and been approved for the course (c.f. G1.3.1).

C8: The system must provide a mechanism to protect student and faculty PII (c.f. G1.4).

C9: A student can only view and update own personal data (c.f. G1.5).

C10: A student can only view own grades (c.f. G1.6)

C11: A faculty member can only view and update own personal data (c.f. G1.7).

C12: A faculty member can only see student data for students enrolled in his or her course (c.f. G1.8).

C13: Official transcripts can only be printed. They cannot be sent via email or saved to portable media. However, an unofficial transcript can be sent via email or be saved to portable media (c.f. G2.5).

C14: The system must allow a student's transition to *alumni* status only after all requirements for graduation have been met (c.f. G5).

C15: Alumni can only view and update own personal data (c.f. G5.1).

C16: The system must provide a mechanism to prevent an authorized user with Registrar permissions from maliciously or accidentally deleting student or alumni records (c.f. G6.1.1). (i.e., two-person integrity on certain transactions)

C17: The system must calculate GPA based on MCU's current procedures (G2.6).

C18: The system must not allow Executive Leadership to modify data (c.f. G4).

C19: The system must provide a mechanism to modify user permissions and granting of authority to privileged users (c.f. G8).

C20: The system must not allow the super user and Registrar to have the same personally identifiable credentials (c.f. G8).

2. Implementation Constraints

C21: The system must interface with existing USMC systems that are actively in use to prevent duplicate data entry and duplicate MCU student registration efforts (c.f. G10).

C22: After a faculty member uploads grades, the system's business rules must be flexible to establish, update, and refine grade approval workflow to ensure the appropriate people can review the grades before they are final (c.f. G1).

C23: The system must address security concerns to allow secure access to RAS from an un-trusted domain (c.f. G1.5, G1.6, G1.7, G1.8).

C24: The system must never allow grades to be frozen. A mechanism must be provided to correct inaccurate data (c.f. G6.2).

O. CONCLUSION: REQUIREMENTS ANALYSIS

In this chapter, a requirements analysis was conducted for an automated RAS for future use at MCU. The refinement of these requirements took place over the course of several months through several telephone conversations, emails between the customer and the analyst, and a user survey. Furthermore, it was an iterative process where the participants reviewed all documents multiple times and communicated their new ideas to each other. This process increased the depth of our understanding of the problem with each iteration. Consensus was achieved; decisions were made and documented. This concludes the requirements analysis portion of this study.

The second part of this study contains a market analysis of the software products that are being used by similar institutions today and a product comparison. It also contains recommendations about which product may be most suitable for MCU based on a series of likely scenarios.

IV. MARKET ANALYSIS

In this section, we introduce institutions similar in nature to MCU as a basis for comparing their software product implementations. Next, we define various product evaluation and comparison methods suitable to evaluate products and aid decision-making when comparing multiple products. We identify the method to be used in this study, and finally we evaluate each software solution against a number of factors.

A. SIMILAR INSTITUTIONS AND ASSOCIATED PRODUCTS

The focus of this market analysis is to discover what software products are in use by MCU's sister institutions and then evaluate them based on common criteria. We intentionally limit the scope of this analysis to the below institutions because of their similar missions, goals, requirements, student and faculty populations, and budgets.

1. Naval War College (Empower)

The Naval War College (NWC) is located in Newport, Rhode Island, and offers professional military education to a mix of 600 mid-career level officers the Navy, all other services, civilian federal agencies, and international naval officers each year [11].

NWC uses a software program called Empower.

2. National Defense University (DES)

The National Defense University (NDU) is located on Fort Lesley J. McNair in Washington, D.C., and offers joint professional military education to U.S. military officers, Department of Defense and other agencies and foreign nations for senior-level policy, command, and staff responsibilities [12].

NDU uses a software program called Data Enterprise System (DES).

3. Air University

Air University (AU) is located on Maxwell Air Force Base, AL, and offers professional military education to officers and employees throughout the Department of Defense and international community [13].

AU has patched together a homegrown conglomeration of software systems to provide services to its students, faculty, and staff. Because this unique mixture of software is not available to implement as a single solution, we will not consider it in our analysis.

4. Army War College (Oasis)

The U.S. Army War College (AWC) is located on Carlisle Barracks in Carlisle, PA, and offers professional development graduate-level education to prepare U.S. military officers, international officers, and senior civilians from throughout government to fill the responsibilities of service in a joint, interagency, intergovernmental, or multinational assignment [14].

AWC uses an ad hoc conglomeration of software systems they refer to as Oasis. The nature of Oasis is not a product that could be packaged up for use elsewhere because of the ad hoc way it strings together and interoperates with AWC's other critical software. Therefore, we will not consider Oasis in this study [15].

5. Naval Postgraduate School (Python)

The Naval Postgraduate School (NPS) is located in Monterey, CA, and offers graduate education to nearly 1,500 from all branches of the U.S. military, DoD, and approximately 30 countries. NPS uses a software program called Python.

6. National University (PeopleSoft)

National University is headquartered in La Jolla, CA, though it has 13 regional campuses throughout California. National University offers 100 graduate and

undergraduate degrees to all diversities of civilian and military students. National University uses a software program called PeopleSoft.

B. EVALUATION AND COMPARISON METHODS

1. Voting

One way to evaluate a set of alternatives is by using voting methods. There are several types of voting methods and each one has advantages and disadvantages that relate to items such as voter abstentions, insincerity, or outcome sensitivity based on where the votes lie. We will discuss three voting methods: approval voting, nominal group voting, and multivoting [16].

These voting methods are important for this study because they can be used iteratively to build up choice criteria that will then be used to formally evaluate multiple products for comparison, pick a vendor or subcontractor, or decide whether to build, reuse, or buy new software [16].

a. Approval Voting

Approval voting requires many voters and allows each voter to vote for as many items as desired. The voter can only vote once for any single item. After tallying the votes, the item receiving the most votes wins. This method is simple to understand and use, however, the number of votes that are cast can very depending upon the voting stakeholders. It is possible for a person to vote for all items in approval voting. This would effectively eliminate that person's vote from impacting the results [16].

Table 1 shows what an approval voting matrix might look like. The features being voted on are listed in the left-most column. There is a listing of each stakeholder in the center section where each stakeholders' votes are represented in a sub column within this section. The quantity of stakeholder votes is accumulated at the base of each of these columns. The right-most column displays the total votes received for each feature [16].

Feature	Stakeholders					Total Votes
	1	2	3	4	5	
A		X				1
B		X	X		X	3
C						0
D	X	X	X	X	X	5
E	X					1
Total Votes By Voter	2	3	2	1	2	

Table 1. Approval Voting Example (From [16])

b. Nominal Group Voting

Nominal group voting requires voters to make a standardized commitment when voting and improves on the sampling of stakeholders of approval voting. An example of a standardized commitment would be to make a rule stating that each voter must cast exactly three votes, or N votes, ranking them in order of importance with one being least important and N being most important. As with approval voting, the voter can only vote once for a single item. If L is the final number of items desired for selection, then limiting the number of votes, N, a person can cast, such that $N < L$, and $N > 0$ will cause the voters to carefully weigh their options before casting their votes. An item that receives no votes gets a score of zero. The L items receiving the highest number of votes are selected and placed on the list [16].

Table 2 shows what a nominal group voting matrix might look like. Features are listed in the left-most column, stakeholders are in the center section, and total votes for each feature are listed in the right-most column. Notice that each stakeholder voted only three times and their votes are numbered in order of ascending importance [16].

Feature	Stakeholder					Total Votes
	1	2	3	4	5	
A		1	2		2	5
B	3	2	3	3	3	14
C						0
D	2	3	1	2	1	9
E	1			1		2

Table 2. Nominal Group Voting Example (From [16])

c. Multivoting

In multivoting, each voter must make a standardized commitment, as in nominal group voting, however, he can vote multiple times for the same item, as opposed to both approval and nominal group voting. All votes may be cast toward a single item if desired. The feature with the most votes wins. Identifying the feature that received votes from the largest amount of unique voters breaks a tie. All other items are then ranked in descending order. Once complete, it becomes easy to pick off your top N features [16].

Table 3 shows what a multivoting matrix might look like. This table adds a new right-most column to display the ranking of the features [16].

Feature	Stakeholder						Total	Rank
	1	2	3	4	5	6	Votes	
A							0	-
B	3	1	1	1	1		7	1
C			1	1		1	3	3
D		1			2		3	4
E		1	1	1		1	4	2
F							0	-
G						1	1	5
H							0	-
Total Votes Cast	3	3	3	3	3	3		

Table 3. Multivoting Example (From [16])

2. Weighted Sum Figure of Merit Analysis

Another way to evaluate between a set of alternatives is to conduct a weighted sum figure of merit analysis. The figure of merit is dimensionless and is calculated for each option. To choose the best option, the option with the higher figure of merit should be selected. The results of using a weighted sum figure of merit are heavily affected by the weights and values manually assigned [17]. This will be the method we employ to compare available products.

We will explore how to do it. Then, we will discuss some concerns with this method and how we will overcome them. For each factor involved in the comparison, start by 1) determining a measurement scale for each factor. Each may be independent from all others [17]. Some examples of Likert performance ratings can be seen in Table 4 and Table 5. For information about the history and development of Likert scales, see "A Technique for the Measurement of Attitudes" by Rensis Likert.

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Never	Seldom	Sometimes	Often	Always
Hourly	Daily	Weekly	Monthly	Yearly

Table 4. Some Performance Ratings (Likert Scales) (From [16])

Very Low	≤ four months
Low	> four months and ≤ one year
Nominal	> one year and ≤ three years
High	> three years and ≤ six years
Very High	> six years

Table 5. Some Performance Ratings (Likert Scales) (From [16])

2) Next, assign a numerical value for each possible outcome on each measurement scale. Table 6 shows an example of how this might look. Note that the measurement scale for training availability is (no, some, full). Also note that the outcome "no" has been assigned the numerical value 1.00 by observing the measurement rating in

the top half of the table and relating it to the assigned value in the bottom half of the table. "Some" has been assigned 2.00 and "full" has been assigned 3.00. Similar measurement scale-to-value mappings have been done for all other factors as well [16].

	G	H	I	J	K	L	M
7	**Factor Name**	**Weight**	**Ratings**				
8	Functionality	0.30	VL	LO	NM	HI	VH
9	Integration Effort (person-weeks)	0.20	1	2	3	4	5
10	Cost (dollars)	0.20	below 500	500 to 1000	above 1000		
11	Vendor Reputation	0.10	Poor	Unknown	Good	Excellent	
12	Product Maturity	0.10	VL	LO	NM	HI	VH
13	Developer Toolkit available	0.05	No	Yes			
14	Training availability	0.05	No	Some	Full		
15		0.00					
16							
17	**Factor Name**		**Values**				
18	Functionality		1.00	2.00	3.00	4.00	5.00
19	Integration Effort (person-weeks)		1.00	2.00	3.00	4.00	5.00
20	Cost (dollars)		1.00	0.00	-1.00		
21	Vendor Reputation		-5.00	-3.00	0.00	9.00	
22	Product Maturity		1.00	2.00	3.00	4.00	5.00
23	Developer Toolkit available		0.00	1.00			
24	Training availability		1.00	2.00	3.00		
25							

Table 6. User-Specified Factors, Weights, and Values (From [16]).

3) Assign factor weights. Generally these weights should add up to one hundred [17]. A variation of this method in [16] sums all the weights up to 1.00. These weights reflect the relative importance of each factor. 4) The next step is to calculate the weighted sum[1] values for each factor. The factors are listed in the left-most column. 5) Assign these values as scores. 6) To choose the best option, select the option with the highest score. Table 7 illustrates this method. By focusing on row five of Table 7, the reader will observe that four options are being considered. They are A, B, C, and D. The weighted sum calculations have been computed and the resulting scores are listed in row six beneath their respective names. Table 7 does not display the assigned weights, though including them would be helpful during results analysis. We see that option D is the best choice at 2.30, A is second best with a score of 1.80. Option C closely follows it. Option B has the lowest score [16].

[1] See [16] for details about how to calculate a weighted sum.

	A	B	C	D	E
1	**Description:**	Evaluation of Four COTS products			
2	**Prepared By:**	Mary Smith			
3	**Date Prepared:**	14-Oct-04			
4					
5	**Option =**	**A**	**B**	**C**	**D**
6	**Score =**	**1.80**	**1.30**	**1.75**	**2.30**
7	**Factor**				
8	Functionality	HI	NM	LO	VH
9	Integration Effort (person-weeks)	1	2	3	4
10	Cost (dollars)	500 to 1000	below 500	500 to 1000	above 1000
11	Vendor Reputation	Good	Poor	Good	Unknown
12	Product Maturity	NM	LO	HI	NM
13	Developer Toolkit available	No	Yes	No	Yes
14	Training availability	Some	No	Full	Full
15					

Table 7. Weighted Scores of Four Products (From [16])

a. Concerns With Weighted Sum Figure of Merit

Reference [17] explains concerns that must be considered before this method should be used to select a product. These concerns are: 1) that incorrect factors and weights may be used. Because this technique ranks factors relative to other factors, one must ensure that the most important factors are included and given appropriately heavy weights. The opposite is true for weights assigned to factors of little importance [17]. 2) That because this method uses an additive function its results may not accurately reflect reality when a factor has a multiplicative impact on system performance, for example [17]. 3) That certain stakeholders might inject their biases by influencing the weight and rating assigned to a factor of their particular interest to increase its weighted impact on the results [17]. This is an interesting technique, which we will later exploit "for the good of our study" to allow the results to appropriately map to three different implementation scenarios.

b. Overcoming the Concerns

Regarding the first concern, we will be particularly careful when conducting this analysis by selecting factors that have key implications in choosing the right product. We will be careful to include important factors and to weight all factors appropriately.

Regarding the second concern, we have concluded that the factors we will use do not have a multiplicative impact on system performance.

Regarding the third concern, we will take special precautions to ensure the biases of stakeholders are not injected into our analysis.

C. EVALUATION OF PRODUCTS

In this study we will evaluate Empower, DES, Python, and PeopleSoft. Air University and the Army War College use an ad hoc conglomeration of software systems that will not be beneficial for us to evaluate here. These four products meet MCU's goals and constraints we developed during requirements analysis.

1. Factors

The factors we will be using to evaluate these products are functionality, cost (in thousands of dollars), vendor/product reputation, product maturity, developer toolkit availability/extensibility, and training availability [16].

Functionality: Each product must meet or be capable of meeting all functional requirements identified during requirements analysis in order to be involved in our product comparison. While every product evaluated below meets the minimum requirements, some may do so to a greater degree than others.

Cost: Cost will be received from the vendor based upon an unofficial request for an estimate because we are not announcing a request for proposal (RFP) to vendors of these products. The estimates herein contained in no way communicate a firm offer or commitment by a vendor. Each cost estimate is based on the two factors: license fee and annual support fee for access to maintenance updates and patches. Due to time and resource restrictions and for the purpose of uniformity we were not able to consider other aspects of cost such as implementation cost, hosting cost, hardware cost, or training cost.

Vendor/Product reputation: Vendor/Product reputation will be determined based on openly available information and reports or studies previously conducted.

Product maturity: Product maturity will be determined from historical data gathered that informs us about a product's age, if it is being actively improved, and software update frequency.

Developer toolkit availability/Extensibility: The availability of a developer's toolkit will be provided by the vendor and will indicate the presence or absence of tools useful to an in-house programmer to customize the product to fit an organization's unique needs. This factor indicates that a product is extensible or not. Extensibility has to do with authority and availability for a customer to expand the product so it is capable of communicating with unique third party software systems already being used by the customer (i.e., by defining interfaces that allow it to communicate with a legacy system that contains critical information).

Training availability: Availability of training will be provided by the vendor and will indicate the level of training that is available to the customer and hosted by the vendor to bring basic users or in-house trainers to a level of proficiency sufficient for the organization.

2. Likert Scales

We assign Likert measurement scales for each of these factors as follows: Functionality (very low, low, nominal, high, very high), Cost[2] (below 250, 250 to 750, above 750), Vendor/Product reputation (Poor, Unknown, Good, Excellent), Product maturity (very low, low, nominal, high, very high), Developer Toolkit availability (no, yes), Training availability (No, Some, Full) [16].

[2] In thousands of dollars.

3. Product Evaluations

a. Empower

We evaluated Empower as indicated in Table 8. A more detailed description that contains the information we considered during our product evaluation can be found below the table.

Factor	Rating
Functionality	Very High
Cost (in dollars)	Below 250
Vendor/Product Reputation	Excellent
Product Maturity	Very High
Developer Toolkit Availability/Extensibility	Yes
Training Availability	Full

Table 8. Empower Evaluation

The data collected and considered during the evaluation process was gathered through private communication with Jon Pilon, Special Markets Director for EMPOWER Student Information System and the Empower product catalog.

Functionality: We rate Empower functionality as "Very High." Empower meets and exceeds all functional requirements developed during Requirements Analysis. Specifically, the following Empower modules would be employed to meet the requirements: Program Manager, Admissions, Records and Registration, Degree Audit, Higher Education Human Resources, Alumni, and Web Portal [18].

Cost: We rate Empower's cost as "below 250." This rating is based on an estimate contained in [18] that lists Empower Software license cost at $66,000 and annual maintenance and support costs at approximately $42,000 for each of the first four years. This totals to approximately $108,000.

Vendor/product reputation: We rate Empower's reputation as "Excellent." Approximately 70 institutions have chosen Empower as their student information system since 2000. These are institutions of various kinds ranging from two-year colleges to professional schools to American tribal colleges [18]. Viewing testimonials from universities such as Holy Cross College, Greensboro College, and Mercy College of Northwest Ohio reveal comments of enthusiasm and praise in support of the Empower product and superior professionalism and competence of the Empower staff [19].

Product maturity: We rate Empower's maturity as "Very High." Development of Empower began in 1995, which makes it approximately 17 years old. Empower is continuously being developed and improved. ComSpec International, Inc. releases four software updates each year, on average, or sooner if needed [18].

Developer toolkit available (extensibility): We rate Empower's developer toolkit availability/extensibility as "Yes." Empower is extensible and provides tools that allow communication with third party systems through creation of interfaces [20].

Training availability: We rate Empower's training availability as "Full." Empower has a full variety of training available in a format suitable for train-the-trainer and individual user settings, both on-site and off-site. Training is also conducted with technical and support staff and reports developers [20].

b. DES

We evaluated DES as indicated in Table 9. A more detailed description that contains the information we considered during our product evaluation can be found below the table.

Factor	Rating
Functionality	Very High
Cost (in dollars)	Below 250
Vendor/Product Reputation	Excellent
Product Maturity	Nominal
Developer Toolkit Availability/Extensibility	Yes
Training Availability	None

Table 9. DES Evaluation

The data collected and considered during the evaluation process was gathered through private communication with Dave Evans, NDU Application Development Support Branch Chief.

Functionality: We rate DES's functionality as "Very High." DES meets and exceeds all MCU requirements. DES contains modules that support institutional research and assessments. Students can rate their peers, faculty, and courses. Assessment reports are easily generated [21].

Cost: We rate DES's cost as "below 250." A DES license would be offered to MCU free of charge. The software is not proprietary and is government property. There would be no annual support fee for DES. NDU would establish a memorandum of understanding (MOU) with MCU to describe that the software comes with no product support capability or assistance provided by NDU [22].

Vendor/product reputation: We rate DES's reputation as "Excellent." This rating is based on the unanimous decision cited in [23] that NDU would abandon all other efforts to explore other student information systems and focus solely on using DES. This decision included votes from all NDU Academic deans, Chief of Staff groups, college Commandants, Directors, and the NDU President [23].

Product maturity: We rate DES's maturity as "Nominal." DES development began in 2002 and has been actively developed since then. Between 2002

and 2008, NDU had been experimenting with DES and portions of the PeopleSoft student information system to determine their suitability and feasibility for NDU. Only in April 2008 did NDU decide to focus solely on refinement and full usage of DES. In 2010, NDU began modifying DES so it runs on a .NET framework instead of using active server page (ASP) technology [23], [22].

Developer toolkit available (extensibility): We rate DES's developer toolkit availability/extensibility as "Yes." DES uses a SQL database and is capable of integrating with existing software system through custom interfaces [21], [22].

Training availability: We rate DES's training availability as "None." NDU does not have the capability or the mission mandate to provide training to outside organizations that desire to implement the DES software. Language expressing this training limitation would be spelled out in the MOU between NDU and the receiving institution [22].

c. Python

We evaluated Python as indicated in Table 10. A more detailed description that contains the information we considered during our product evaluation can be found below the table.

Factor	Rating
Functionality	Very High
Cost (in dollars)	Below 250
Vendor/Product Reputation	Good
Product Maturity	High
Developer Toolkit Availability/Extensibility	Yes
Training Availability	Full

Table 10. Python Evaluation

The data collected and considered during the evaluation process was gathered through private communication with Mike Andersen, NPS Registrar, and Keith Jones, Director of Application Support with eDataTech.

Functionality: We rate Python's functionality as "Very High." Among other functional capabilities, Python is already suited to break out students according to military branch, government agency, or international country demographics. Python uses familiar military terminology. Python already operates on a demand-based scheduling foundation, which is different from most civilian institutions. Demand-based scheduling is essential when the university establishes firm graduation dates [24].

Cost: We rate Python's cost as "below 250." A Python license would be offered to MCU free of charge because it was developed by NPS students and belongs to the government. Typically, the need to build customized reports and to modify Python to accommodate institutional evolutionary events have driven annual maintenance and support costs for NPS. Some of these events might be moving into the distance learning domain or hosting nontraditional students such as government civilians and contractors. If implemented, a support contract would be negotiated and would probably cost less than $200,000 annually. It is estimated that an annual support fee would cost MCU approximately $200,000 the first year, $150,000 the second year, and $120,000 the third year and following years [24].

Vendor/product reputation: We rate Python's reputation as "Good." Python's user interface is a bit outdated; it is built using ASP technology [25]. Python's reputation is evaluated as "good" based on class notes reflecting discussion and instructor input during a Human Computer Interaction course [26].

Product maturity: We rate Python's maturity as "High." Python is nine years old and has been continuously been developed and enhanced since then. During fiscal year 2010, there were 32 data fixes and 23 new features applied to the system addressing 47 bug reports [27].

Developer toolkit available (extensibility): We rate Python's developer toolkit availability/extensibility as "Yes." Python runs on a SQL database and is

compatible with Oracle database software. Python allows for interfaces to be defined allowing it to communicate with third-party software [25].

Training availability: We rate Python's training availability as "Full." A full range of training would be available to MCU through NPS personnel highly experienced with Python that could provide training on all aspects of usage to include students, faculty, and Registrar staff [24].

d. PeopleSoft

We evaluated PeopleSoft as indicated in Table 11. A more detailed description that contains the information we considered during our product evaluation can be found below the table.

Factor	Rating
Functionality	High
Cost (in dollars)	Below 250
Vendor/Product Reputation	Poor
Product Maturity	Very high
Developer Toolkit Availability/Extensibility	Yes
Training Availability	Full

Table 11. PeopleSoft Evaluation

The data collected and considered during the evaluation process was gathered through private communication with Frances Schreiner, an Oracle Higher Education California Sales Representative, Oracle's public web pages, private communication with Gary Humphrey of Cedar Crestone, private communication with Mike Andersen, NPS Registrar, a business case report published by NDU in 2008, and private communication with Brent Kelley, Oracle Sales Consulting Manager. Cedar Crestone is third party whose business includes technical experts trained in

implementation and maintenance of PeopleSoft solutions. We consider Mr. Andersen's input of value because of his past analysis of PeopleSoft as a potential student information system solution for NPS.

Functionality: We rate PeopleSoft's functionality as "High." PeopleSoft is a highly advanced product and comes with much functionality. PeopleSoft is so advanced that it provides excess features to a fault when being considered for an institution like Marine Corps University. Much functionality that comes with it does not apply to MCU and will not be used. Because of this, we rate it as "high" in this category [24].

Cost: We evaluate PeopleSoft's cost as "Below 250." This is based on a cost estimate provided by [28]. The estimated total cost for a PeopleSoft Enterprise Student Administration software license and annual support is $191,795. This amount is broken down into two components. They are Net License at $157,200 and Net Support at $34,595. These values are based on 1000 system users at MCU and would necessarily require adjustment as the number of licenses needed fluctuates [28].

Vendor/product reputation: We evaluate PeopleSoft's reputation as "Poor" though it is used at over 800 campuses in more than 20 countries [29]. Feedback regarding PeopleSoft was very negative in a business case that analyzed it for continued use at NDU. To name a few issues, usability was so bad that NDU has lost prospective students because they had given up when trying to navigate the application screen. Senior officers have found the many screens required to enter biographical data as troublesome and refuse to enter their data. PeopleSoft's ability to provide historical information is incomplete [23].

Product maturity: We rate PeopleSoft's maturity level as "Very High." PeopleSoft is continually being developed, improved, and updated. All the modules that are offered are integrated with each other [30].

Developer toolkit available (extensibility): We rate developer toolkit availability/extensibility as "Yes." PeopleSoft Enterprise PeopleTools 8.51 is available [31].

Training availability: We rate training availability as “Full.” There is much training available through Oracle University covering all facets of PeopleSoft Campus Solutions [32].

V. PRODUCT COMPARISON

In this chapter, we will map our Likert rating scales to numerical values. Then, we will define the three most probable scenarios in which MCU might find itself. For each scenario, we will assign weights to each factor to simulate the reality of that scenario and calculate the weighted sum for each product. Finally, we will recommend the best product for each scenario to MCU for consideration and future decision-making.

A. LIKERT SCALE RATING-TO-VALUE MAPPING

We map values to the rating scale used for each factor as indicated in Table 12:

Factor	Rating to Value Mapping					
Functionality	Rating	Very Low	Low	Nominal	High	Very High
	Value	1.00	2.00	3.00	4.00	5.00

Cost	Rating	below 250	250 to 750	above 750
	Value	1.00	0.00	-1.00

Vendor/Product Reputation	Rating	Poor	Unknown	Good	Excellent
	Value	-5.00	-3.00	0.00	3.00

Product Maturity	Rating	Very Low	Low	Nominal	High	Very High
	Value	1.00	2.00	3.00	4.00	5.00

Developer Toolkit Availability / Extensibility	Rating	No	Yes
	Value	0.00	1.00

Training Availability	Rating	None	Some	Full
	Value	1.00	2.00	3.00

Table 12. Rating-to-Value Mapping (From [16])

B. MCU'S THREE MOST PROBABLE SCENARIOS

A constant within the military is that the situation and environment are always changing. MCU is an institution whose surrounding environment ebbs and flows with

the needs of the Marine Corps. Because of this dynamic environment, we are hesitant to pick only a single software product to recommend to MCU. Rather, we want to define three most probable scenarios in which MCU might find itself. We did this by working directly with MCU to ensure our scenarios present realistic possibilities and to remove an amount of subjectivity from this study. Therefore, MCU provided the scenarios we use below. Then, based on each of those scenarios, we will recommend a product that fits best. In the end, MCU will be presented with three courses of action from which it can choose to best satisfy its needs. These scenarios are important because they will aid MCU when deciding between these products in the future.

1. Scenario 1, Maximum Capability

We define Scenario 1 where MCU is not resource constrained and can purchase a product with the greatest capability and potential for future expansion. MCU is able to fund licensing fees and support personnel to provide a product that delivers maximum capability to accomplish the mission [33].

2. Scenario 2, In House System

We define Scenario 2 where MCU has funding to provide organic manpower to customize and maintain the system, however, it does not have significant funding for high startup costs. This option will avoid payment of expensive vendor-provided maintenance costs and allow installation and maintenance to be done "in house." A product purchased for this scenario would cost less up front, but have a significant cost to support and maintain the product over time [33].

3. Scenario 3, Commercial Product with Contractor Support

We define Scenario 3 as a hybrid of the first two scenarios. MCU's circumstances dictate that it is better for the institution to license the product and purchase a maintenance contract from the vendor. MCU is not able to maintain the system "in house" because a maintenance contract is more cost effective than the salaries of organic support personnel [33].

C. WEIGHTING THE SCENARIOS

After the scenarios were defined we, again, sought input from MCU. To remove further subjectivity from this study and because MCU best knows its own priorities, we asked MCU to determine weighting values for the six factors according to each factors' relative importance to the institution. MCU provided these weighting values for each scenario. These weighting values provide a mechanism to simulate the reality of the scenario to which they belong.

For example, for Scenario 1, assignment of a lesser weight to "cost" allows the other factors to have a higher impact on the result. For Scenario 2, assigning a greater weight to "developer toolkit availability" simulates MCU's ability to hire support personnel.

Adjusting the weights appropriately allows the results to take shape around the specific scenarios and will more accurately inform us which product is best to pursue in each scenario.

	Scenario 1	Scenario 2	Scenario 3
	Maximum Capacity	**In House System**	**Commercial Product with Contractor Support**
Factor	weighting value	weighting value	weighting value
Functionality	0.4	0.15	0.25
Cost (in dollars)	0.05	0.3	0.35
Vendor/Product Reputation	0.15	0.1	0.15
Product Maturity	0.2	0.1	0.15
Developer Toolkit Availability / Extensibility	0.05	0.3	0
Training Availability	0.15	0.05	0.1
	1	1	1

Table 13. Scenario Weighting Values (From [33])

D. GATHERING THE PARTS TO MAKE THE WHOLE

In Chapter IV, we evaluated each product. In this chapter, we established a mapping between the Likert ratings and numerical values. Through communication with MCU, we defined three scenarios and appropriate weighting values corresponding to each scenario. This section of our study incorporates all of these parts and will allow us to see which product is best for each scenario.

Each scenario is represented below with a table. The table displays the six factors in the left-most column, the MCU-assigned weights in the next, and the assigned values for each of our four products in the remaining columns. The bottom row in each table displays the weighted sum for each product.

1. Scenario 1, Maximum Capability

Factor	Weight	Empower	DES	Python	PeopleSoft
Functionality	0.4	5.00	5.00	5.00	4.00
Cost (in dollars)	0.05	1.00	1.00	1.00	1.00
Vendor/Product Reputation	0.15	3.00	3.00	0.00	-5.00
Product Maturity	0.2	5.00	3.00	4.00	5.00
Developer Toolkit Availability / Extensibility	0.05	1.00	1.00	1.00	1.00
Training availability	0.15	3.00	1.00	3.00	3.00
Weighted Sum		4.00	3.30	3.35	2.40

Table 14. Scenario 1 Weights, Ratings, and Weighted Sums

2. Scenario 2, In House System

Factor	Weight	Empower	DES	Python	PeopleSoft
Functionality	0.15	5.00	5.00	5.00	4.00
Cost (in dollars)	0.3	1.00	1.00	1.00	1.00
Vendor/Product Reputation	0.1	3.00	3.00	0.00	-5.00
Product Maturity	0.1	5.00	3.00	4.00	5.00
Developer Toolkit Availability / Extensibility	0.3	1.00	1.00	1.00	1.00
Training availability	0.05	3.00	1.00	3.00	3.00
Weighted Sum		2.30	2.00	1.90	1.35

Table 15. Scenario 2 Weights, Ratings, and Weighted Sums

3. Scenario 3, Commercial Product with Contractor Support

Factor	Weight	Empower	DES	Python	PeopleSoft
Functionality	0.25	5.00	5.00	5.00	4.00
Cost (in dollars)	0.35	1.00	1.00	1.00	1.00
Vendor/Product Reputation	0.15	3.00	3.00	0.00	-5.00
Product Maturity	0.15	5.00	3.00	4.00	5.00
Developer Toolkit Availability / Extensibility	0	1.00	1.00	1.00	1.00
Training availability	0.1	3.00	1.00	3.00	3.00
Weighted Sum		3.10	2.60	2.50	1.65

Table 16. Scenario 3 Weights, Ratings, and Weighted Sums

E. BEST PRODUCT FOR SCENARIO 1

If MCU is closest to Scenario 1, maximum capability, Empower comes through as the best product to pursue. Second place is Python with DES a close third.

F. BEST PRODUCT FOR SCENARIO 2

If MCU is closest to Scenario 2, in house system, Empower comes through as the best product to pursue. Second place is DES with Python in third place.

G. BEST PRODUCT FOR SCENARIO 3

If MCU is closest to Scenario 3, commercial product with contractor support, Empower comes through as the best product to pursue. Second place is DES with Python in third place.

H. SUMMARY

In this chapter we mapped our Likert rating scales to numerical values. We defined the three most probable scenarios in which MCU might find itself. For each scenario, we assigned weights to each factor to simulate the reality of that scenario and calculated the weighted sum for each product. Finally, we recommended the best product for each scenario to MCU for consideration and future decision-making.

For each scenario, Empower is the product that will best meet MCU's needs. For a system with maximum capability, Scenario 1, Python is second best. For an In-House system or a commercial product with contractor support (Scenarios 2 and 3), DES is second best. In all three scenarios, PeopleSoft is recommended as a last resort.

VI. CONCLUSION AND FUTURE WORK

A. CONCLUSION

Chapter II laid a foundation for RA. We gave attention to methods that described one way to conduct RA. Specifically, we discussed how to develop the initial problem statement, environment model, goals, goal hierarchy, and constraints.

In Chapter III, we put these methods into practice by conducting a RA for our customer, MCU. We developed a set of requirements based on the needs of MCU for such an RAS. We composed an initial problem statement, environment model, goals, and constraints. We built high-level use cases, used design notation and context diagrams, and examined external interfaces to help us gain a better understanding of the problem domain. We conducted a user survey to gather input across the range of users. We considered this input and communicated with MCU to refine our goals, constraints, and environment model. Chapter III concluded with a completed goal hierarchy and list of constraints. After we established MCU's requirements, we conducted a business study to evaluate and compare products currently in use.

In Chapter IV, we conducted a market analysis and product evaluation. We started with a market analysis to learn about the systems that are being employed at institutions similar to MCU. We discussed various evaluation and comparison methods and then chose one of them to employ for our own use. Next, we defined the product characteristics, or factors, we would consider and Likert rating scales we intended to use in our evaluation. We concluded Chapter IV by evaluating four candidate software products.

In Chapter V, we conducted a product comparison that was based on the evaluations we conducted in Chapter IV. We started by mapping our Likert rating scales to numerical values. Next, we defined three most probable scenarios in which MCU might find itself in the near future. We established weighting values for each factor based upon each of the three scenarios. We then brought our product evaluations,

scenarios, and weighting values together and compared the products by calculating weighted sums. Empower emerged as the best product for each scenario. Python emerged as the second best product for Scenario 2. DES emerged as the second best product for Scenarios 2 and 3.

B. RECOMMENDATIONS FOR FUTURE WORK

The following ideas are possible areas where future work would be done.

1. Other MCU Capability Gaps

MCU is still a relatively young organization and is working to improve itself in many areas. Recommend the reader connect with MCU's IET Director to discuss potential topics of study related to MCU.

2. MCU Unity of Command topics

Work could be done to further study unity of command as it relates to curriculum development, content delivery, infrastructure, network management, or IT procurement. One could also explore solutions to any of the other illuminated concerns in [4], such as reports development.

3. Kuali Student Evaluation

Work could be done to research Kuali Student as a possible student information system solution. Kuali Student is presently in the very early stages of development and will be an open-source solution.

4. Software Fit to MCU

Work could be done to analyze the "fit" of student information systems to meet MCU's unique student and academic needs.

5. RAS Design Development

Work could be done to take the RAS requirements analysis conducted here and apply the in vogue rapid prototyping development method to produce a usable product for MCU. The method, language, process, and tools are described in detail in the references along with a case study on a command and control system. Prototyping is an adaptive approach that solidifies and validates requirements in a "build a little-test a little" process [34], [35], [36], [37]. Recommend the reader connect with NPS Professor Luqi at luqi@nps.edu.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] Marine Corps University, *Factbook 2010-2011,* MCU Institutional Research Assessment & Planning & Marine Corps University Press, 2010.

[2] *Marine Corps University History*. (2009, May). MCU President's Document Library. Retrieved from http://www.mcu.usmc.mil/Presidents%20Document%20Library/MCU%20History2.pdf.

[3] C. E. Wilhelm, W. C. Gregson, B. B. Knutson, P. K. Van Riper, A. F. Krepinevich, and W. Murray, "U.S. Marine Corps Officer Professional Military Education 2006 Study and Findings," Marine Corps University, Quantico, Virginia, 2006.

[4] E. D. Ferris, J. L. Terry, H. Villarama, and B. Armstrong, "Marine Corps University Information & Education Technology Master Plan," Flatter and Associates, Quantico, Virginia, Solicitation Number N00024-07-R-3490, 2008.

[5] V. Berzins, & Luqi. *Software Engineering with Abstractions.* Reading, MA: Addison-Wesley Publishing Company, Inc., 1991.

[6] R. S. Pressman, *Software Engineering: A Practitioner's Approach*. Boston: McGraw-Hill, 2010.

[7] A. I. Anton, Goal-based requirements analysis," in Proceedings of the Second International Conference on Requirements Engineering, 1996, pp. 136–144.

[8] A. M. Davis, *Software Requirements: Objects, Functions, & States*. Prentice-Hall, 1993.

[9] F. P. Brooks Jr., *The Design of Design: Essays from a Computer Scientist*. Indianapolis: Addison-Wesley, 2010.

[10] Luqi, "Software evolution through rapid prototyping," *IEEE Computer*, vol. 22, no. 5, pp. 13–25, May 1989.

[11] Public Affairs Office, Naval War College, "Academics Overview," April 6, 2011, http://www.usnwc.edu/Academics.aspx.

[12] Public Affairs Office, National Defense University, "Factsheet: AY 2009-2010," April 6, 2011, http://www.ndu.edu/info/NDU%20Factsheet.pdf.

[13] Public Affairs Office, "Air University Facts Sheet," April 6, 2011, http://www.au.af.mil/au/facts.asp.

[14] Public Affairs Office, U. S. Army War College, “U.S. Army War College Program Overview,” April 6, 2011, http://www.carlisle.army.mil/usawc/about/programOverview.cfm.

[15] T. Flynn (private communication), 2011.

[16] R. D. Stutzke, “Tools for decision analysis and resolution,” *Crosstalk: The Journal of Defense Software Engineering*, vol. 20, no. 11, pp. 23–26, November 2007.

[17] B. W. Boehm, *Software Engineering Economics.* Englewood Cliffs NJ: Prentice-Hall, 1981.

[18] J. Pilon (private communication), 2011.

[19] ComSpec International, Inc., *Empower, Software Making Higher Education Work*, Bingham Farms, MI, 2010.

[20] R. H. Brender, “Empower Student Information System,” ComSpec International, Inc., Naval War College Source Selection Information Tech. Rep. N00189-08-T-Z071, 2008.

[21] National Defense University Information Paper, *Data Enterprise System (DES)*, Washington, D. C., 2011.

[22] D. Evans (private communication), 2011.

[23] K. Hogan, “Business case to support uNet[PeopleSoft]/DES recommendation,” National Defense University Information Technology Steering Committee, 2008.

[24] P. Andersen (private communication), 2011.

[25] K. Jones (private communication), 2011.

[26] “Human Computer Interaction,” class notes for CS3004, Department of Computer Science, Naval Postgraduate School, Fall 2011.

[27] A. Pires (private communication), 2011.

[28] B. Kelley (private communication, 2011.

[29] Oracle, “PeopleSoft Enterprise Campus Solutions,” April 2011, http://www.oracle.com/us/products/applications/peoplesoft-enterprise/campus-solutions/052356.html.

[30] G. Humphrey (private communication), 2011.

[31] Oracle, “PeopleSoft Enterprise Tools and Technology,” April 2011, http://www.oracle.com/us/products/applications/peoplesoft-enterprise/tools-tech/index.html.

[32] Oracle University, “Campus Solutions Training,” April 2011, http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getSearchResults?p_search_category_id=1186&p_org_id=1001&p_lang=US.

[33] W. Wright (private communication), 2011.

[34] Luqi, “Computer-aided software prototyping,” *IEEE Computer*, vol. 24, no. 9, pp. 111–112, September 1991.

[35] Luqi, “Computer-aided prototyping for a command-and-control system using CAPS,” *IEEE Software*, vol. 9, no. 1, pp. 56-67, January 1992.

[36] Luqi, G. Jacoby, “Testing adaptive probabilistic software components in cyber systems,” Submitted to Springer LNCS, 2010.

[37] Luqi, F. Kordon, “Modeling, development, and verification of adaptive systems,” in *Proceedings of the 16th Monterey Workshop*, 2010, pp. 1–42.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor Luqi
   Computer Science Department
   Naval Postgraduate School
   Monterey, California

4. Dr. Karl Pfeiffer
   Information Sciences Department
   Naval Postgraduate School
   Monterey, California

5. Marine Corps Representative
   Naval Postgraduate School
   Monterey, California

6. Director, Training and Education, MCCDC, Code C46
   Quantico, Virginia

7. Director, Marine Corps Research Center, MCCDC, Code C40RC
   Quantico, Virginia

8. Marine Corps Tactical Systems Support Activity (Attn: Operations Officer)
   Camp Pendleton, California

9. Mr. William Wright
   Marine Corps University
   Quantico, Virginia

10. Mr. Richard Jaques
    Marine Corps University
    Quantico, Virginia

11. Daniel Good
    Manpower Information Technology (MIT) Division
    Quantico, Virginia

12. Dan Boger
    Naval Postgraduate School
    Monterey, California